1995年9月1日発行（毎月1回1日発行）第14巻9号通巻161号 昭和58年11月2日第三種郵便物認可

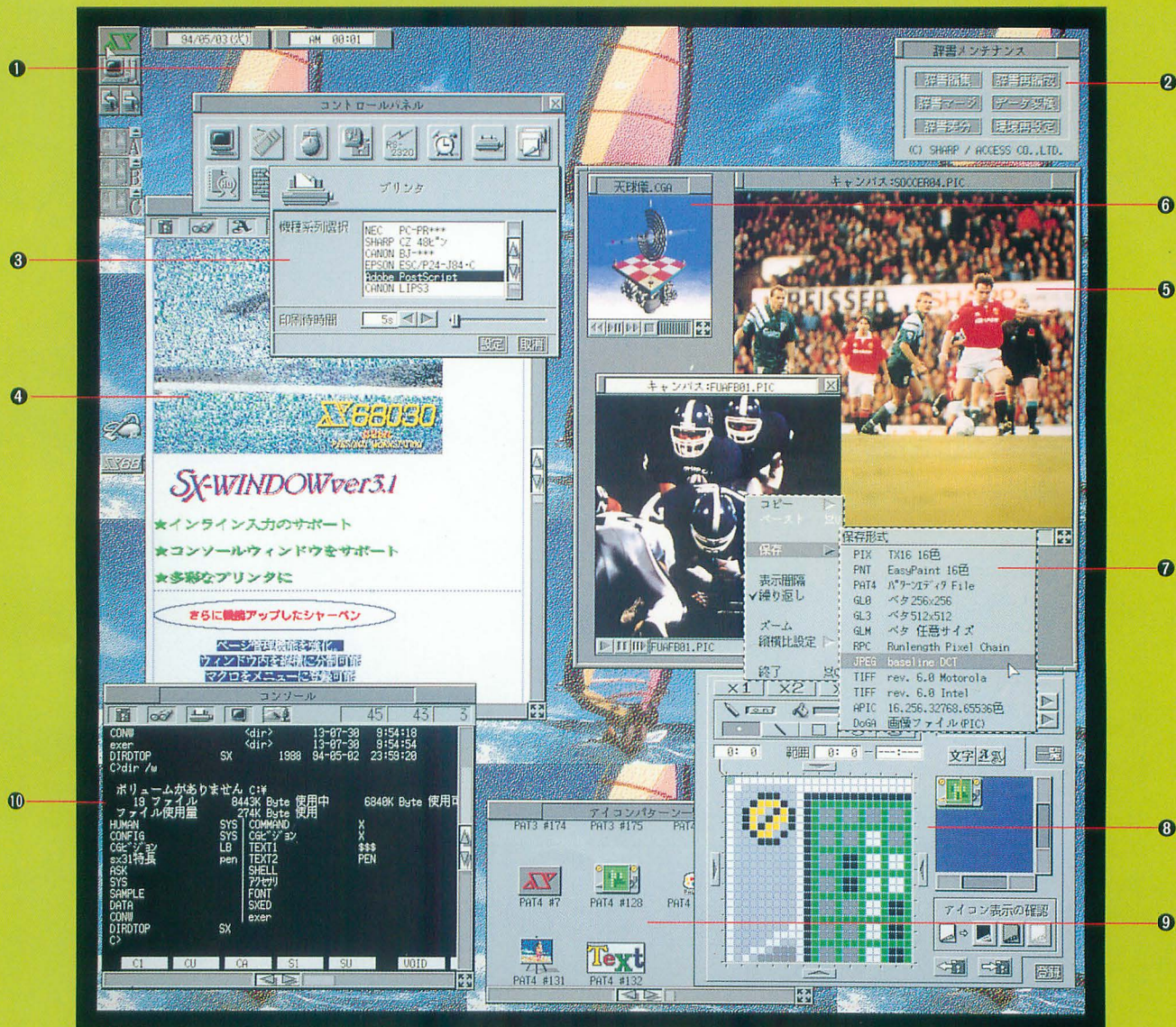
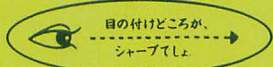
PERSONAL COMPUTER MAGAZINE for MZ, X1, and X68000

Oh! X

特集 Animation Now !

SCSI 2を使用したHDアニメーション／シネパックのアルゴリズムを見る
AMIデータ加工ツール／SCSI2ボードの可能性／SCSIによる究極の動画像環境

1995

SOFT BANK
オー／エックス
定価760円

■実画面：1,024×1,024ドット、表示画：768×：512ドット

●画面は広告用に作成した、機能を説明するためのイメージ画面です。また、各種アイコンなどは、SX-WINDOW ver.3.1がもつ機能を使って作成したもので、標準装備のものとは異なるものもあります。
●本広告中の「シャーペン」で表示している文字のフォントはツァイト社の、「書体倶楽部」のフォントを使用しています。

❶「パターンエディタ」で作成したデータを背景に設定可能。

❷日本語フロントプロセッサ ASK68K ver.3.0の辞書メンテナンスがウィンドウ上で可能。

❸ESC/Page、LIPSIII、PostScriptに対応したプリンタが利用できます。

❹付属アプリケーション「シャーペン」編集例。文字ごとに文字種・文字の大きさの指定、装飾が可能。またインライン入力をサポート、イメージデータの貼りつけもOK。

❺512×512ドットの範囲内で65,536色の表示が可能。

❻「CGAウィンドウ」、65,536色（最大）のコンピュータアニメーション表示が可能。

❼異なる画像フォーマットへのコンバートが可能。

❽アイコンデータや背景データを作成する「パターンエディタ」。

❾オリジナルに作成したアイコンパターンの例。

❿Human68kやX-BASICのコマンドをSX-WINDOWアプリケーションと同時にタイムシェアリングで実行できます。

■「SX-WINDOW ver3.1システムキット」のお問い合わせは…シャープ㈱電子機器事業本部（液映）システム機器推進プロジェクトチーム　〒162 東京都新宿区市谷八幡町8番地 ☎(03)3260-1161（大代表）へ

フィールドが、膨らむ。

68買ったら EXEクラブへ入ろう!

EXE クラブって何だ?

X68030/X68000を手に入れて、いろいろチャレンジしたい皆さん。情報のチャンネルは多いほどいいですよね。ということで、EXEクラブは68ユーザーのための水先案内人。あなたのチャレンジを強力にバックアップしますよ。

本体同梱の入会申込ハガキを送るだけで、自動的に無料入会。さらに下記の特典付き。

メリット1　会員ナンバー入りのオリジナル会員電卓がもらえる。

メリット2　各種フェアご優待・イベント案内等、数々の特典がある。

先が、ますます面白くなる。

●

未来への確かなビジョンをベースに
発展性のあるプラットホームとしてのウィンドウ環境を提供する
国産オリジナルウィンドウシステムSX-WINDOW。

●

GUI環境や操作環境、高速化へのゆるぎない探求、
マルチメディアの統合的なハンドリング。

●

いま、より多彩なフィールドへ
そのインテリジェンスが展開を始める。

●

次のステージが見えてくる。

●**インライン入力のサポート**：ASK68K Ver.3.0を利用したインライン入力をSX-WINDOWで実行可能。またシャーペン.Xをワープロとして利用できるよう、さまざまな機能が付加されています。

●**コンソールをサポート**：Human68kやX-BASICのコマンドをSX-WINDOWアプリケーションと同時にタイムシェアリングで実行できます。
(グラフィックを利用したものなど、SX-WINDOWと処理が重複するものは実行できません。)

●**多彩なプリンタに対応**：さまざまなSX-WINDOWアプリケーションで利用できるページプリンタドライバを標準装備。ESC/Page、LIPS III、PostScriptに対応したプリンタが利用できます。

SX-WINDOW ver.3.1の
データ利用環境

今も、先も楽しめる。
いつも新展開の予感、SX-WINDOWのニューバージョン。

「SX-WINDOW ver3.1システムキット」CZ-296SS(130mmFD)/CZ-296SSC(90mmFD) 標準価格22,800円(税別)

特集 Animation Now!

THE USER'S WORKS in TAKERU

サイバリオン

X68000ゲーム年代記

OCR.X

(で)のショートプロぱーてぃ

Oh!X

C O N T

●特集

●カラー紹介

●THE SOFTOUCH

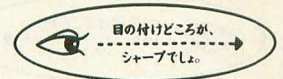
〈スタッフ〉
●編集長／前田 徹 ●副編集長／植木章夫 ●編集／山田純二 高橋恒行 ●協力／有田隆也 中森 章 林 一樹 吉田幸一 華門真人 朝倉祐二 大和 哲 村田敏幸 丹 明彦 三沢和彦 長沢淳博 清瀬栄介 柴田 淳 瀧 康史 横内威至 進藤慶到 菊地 功 伊藤雅彦 ●カメラ／杉山和美 ●イラスト／山田晴久 江口響子 高橋哲史 川原由唯 ●アートディレクター／島村勝頼 ●レイアウト／元木昌子 加藤真二 ●校正／グループごじら

表紙絵：塚田　哲也

1995 SEP. 9

E　N　T　S

■広告目次

ビデオグラフィックスの
世界へ。
■お問い合わせは… シャープ株式会社
電子機器事業本部システム機器営業部 〒545 大阪市阿倍野区長池町22番22号 ☎(06)621-1221(大代表)
資料請求券
X68030

1,677万色対応、ビデオ映像を高画質・高速取り込み

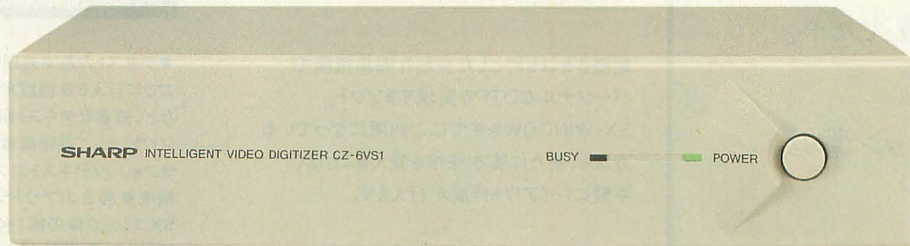
テレビやビデオ、ビデオディスクなどの映像をX68シリーズやMacシリーズ※1の動画・静止画データとして高速取り込みが可能、いわば“ビデオスキャナ”とでも呼びたいビデオ入力ユニットです。1,677万色対応、最大640×480ドットの高解像度※2。動画・静止画の手軽なハンドリングが、新たなグラフィックシーンを創造します。

※1 MacintoshはIIシリーズ以降の機種に対応、ディスプレイ解像度が640×480ドットの場合、取り込み可能な範囲は、160×120ドット、320×240ドットのサイズになります。
※2 X68030/X68000シリーズでは、1,677万色はデータ作成のみに対応。表示は最大65,536色、解像度は512×512ドット。また、Macintoshは機種により表示色数が異なります。

アプリケーションツール「ライブスキャン」を標準装備

動画や静止画を簡単に保存できるアプリケーションソフト「ライブスキャン」※を標準装備。取り込んでいる映像を表示したり、残したいシーンを簡単に静止画保存したり、手軽な動画・静止画ハンドリングでパソコンの可能性をさらに広げます。X68030/X68000シリーズ用SX-WINDOW対応版とMacintoshシリーズ用QuickTime対応版の2種類を同梱しています。

※SX-WINDOW版はバージョン3.0以降(メモリー4MB以上)、QuickTime版はMacintosh漢字Talk7リリース7.1以上のシステムとQuickTime1.5以上(メモリー8MB以上)が必要です。

1,677万色対応の高速映像取り込み、動画・静止画の手軽なハンドリングが、新たなマルチメディアシーンを創造する。

■SCSIインターフェイス採用:パソコンの専用I/Oスロットを使わずに接続可能になり、汎用化を実現しました。またSCSI-2(FAST)インターフェイスの採用により、データ転送速度の高速化を図っています。X68030/X68000シリーズでは、SCSI-2(FAST)対応のハードディスクを接続することにより、パソコン本体を経由しないで、ハードディスクに直接、動画データをテンポラリデータとして記録することが可能です。パソコン本体のハードディスクへは、記録終了後に、テンポラリデータを変換し動画データとして保存できます。

※CZ-600C/601C/611C/602C/612C/652C/662C/603C/613C/653C/663Cに接続する場合は別売のSCSIインターフェイスボードCZ-6BS1ならびにSCSI変換ケーブルCZ-6CS1が必要です。※CZ-604C/623C/634C/644Cに接続する場合は、別売のSCSI変換ケーブルCZ-6CS1が必要です。
※Macintosh Power Bookシリーズに接続する場合は別売のSCSIケーブルなどが必要です。詳しくはMacintosh Power Bookシリーズの取扱説明書をご覧ください。

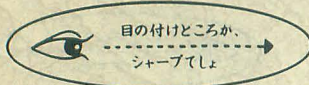
■高機能MPUを搭載:クロック周波数25MHzの32ビットMPU/MC68EC020を搭載、高速処理やパソコン本体の負担の軽減を実現します。

●MacはMacintoshの略称です。●Macintosh、Macintosh IIは、米国アップルコンピュータ社の登録商標です。●Power Bookは米国アップルコンピュータ社の商標です。●漢字Talk7はアップルコンピュータジャパン社の商標です。●QuickTimeは、米国アップルコンピュータ社の商標です。●価格には、消費税及び配送・設置・付帯工事費、使用済み商品の引き取り費等は含まれておりません。

for X68 Mac

ビデオ入力ユニット

CZ-6VS1

標準価格178,000円(税別)

For X68030/X68000series

ORIGINAL SOFTWARE COLLECTION

さらに高度な創造次元へ。
ますます成熟する
そのアプリケーション環境。

X68030
32bit PERSONAL WORKSTATION

NEWアプリケーション

●独自のアウトラインフォントを付属

書家万流 フォント&ロゴ デザインツール SX-68K

CZ-282BWD　標準価格29,800円(税別)

4MB | ver.3.0 | HD 10MB

フォントやロゴを手軽に作成するための
デザインツール。作成したロゴは
クリップボードを介し、シャーペンや
EGWord SX-68K、XDTP SX-68Kなど
他のアプリケーションで利用できます。

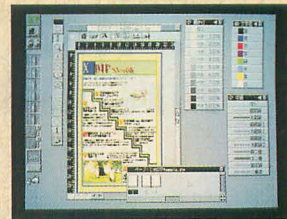
●SX明朝体/SXゴシック体フォント(JIS第1水準&第2水準)を付属●ベジェ曲線のアウトライン編集によるデータ作成●フォントファイル全体にわたってのエフェクト処理●既存のフォントファイルからのデータ抽出、ドローオブジェクトへのエフェクト処理●複数のフォントファイルをリンクして新たなフォントファイルの作成が可能●65,536色表示で確認しながらロゴ作成ができるグラフィックウィンドウ(GRW.X)対応

●パーソナルDTPをX68で

XDTP SX-68K

CZ-291BWD　標準価格35,000円(税別)

4MB | ver.3.0 | HD 5MB

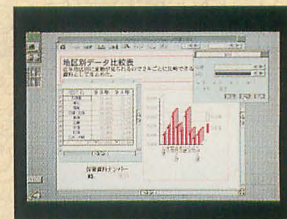
縦書きをはじめとした多彩な編集機能で
パーソナルなDTPを実現するソフト。
SX-WINDOWをすでにご利用になっている
方なら、新たに基本操作を覚えることなく
手軽にレイアウト作成が行えます。

●テキストの基本処理をはじめ、テキストフレームごとに行える各種設定、スタイル別の検索/置換など、豊富なテキスト編集機能●グラフィックウィンドウ、そして各種画像フォーマットへの対応●グラフィック/テキストのフレームから独立した罫線機能●独自のアウトラインフォント(SX明朝体、SXゴシック体の第1水準)標準添付●ページの移動/作成/削除がスピーディに行える独立したページウィンドウをサポート

●DTP感覚で自在にレイアウト編集

Datacalc SX-68K

CZ-273BWD　標準価格59,800円(税別)

4MB | ver.3.0 | HD 3MB

SX-WINDOW対応の新世代統合ソフト。
表計算、グラフ、データベース、テキスト、罫線の
各データを1枚の用紙に重ね合わせ、
移動、サイズ変更など
DTP感覚でレイアウト編集ができます。

●カルクシートでは、セル番地を意識することのない直感的なセル指定が可能●データベースフィールドでは、同一項目でもデータ型/データ長の異なったデータを管理できるなど、自由な設計が特長●データベースフィールドで入力したデータをカルクシートのデータとして利用したり、カルクシートのデータ変更を自動的にグラフ表示に反映させたり、同一データからさまざまな分析が可能なデータリンクもサポート

システム & アプリケーション

●さらに実用的なウィンドウシステムへの進化

SX-WINDOW ver3.1 システムキット

CZ-296SS(130㎜FD)/CZ-296SSC(90㎜FD)　標準価格22,800円(税別)　4MB

ASK68K ver.3.0を利用したインライン入力のサポート、Human68k/BASICコマンドをSX-WINDOWアプリケーションと同時にタイムシェアリングで実行できるコンソールのサポートをはじめ、シャーペン、Xをワープロとして利用できるよう機能アップ。また、さまざまなSX-WINDOWアプリケーションで利用できるページプリンタドライバを標準装備。ドローデータ(FSX)/フォントデータ(IFM)処理の高速化も実現しています。

※コンソールでは、SX-WINDOWと処理が重複するものは実行できません。

●SX-WINDOWを楽しく使うためのアクセサリ集

SX-WINDOW デスクアクセサリ集

CZ-290TWD　標準価格14,800円(税別)

SX-WINDOWをさらに便利に、楽しく使うためのデスクアクセサリ集です。スクリーンセーバ、スクラップブック、アドレス帳、電子手帳通信ツールなど、12種の豊富なアクセサリが収められています。

4MB | ver.3.0

●SX-WINDOW対応ドローイングツール

Easydraw SX-68K

CZ-264GWD　標準価格19,800円(税別)　4MB | ver.3.0

イラスト、フローチャート、地図、見取り図など各種グラフィックが製図感覚で作成できます。作成したデータは他のSX-WINDOW対応アプリケーションでも利用でき、企画書などの作成をサポートします。

●ウィンドウ対応のグラフィックツール

Easypaint SX-68K

CZ-263GWD　標準価格12,800円(税別)　2MB | ver.1.1

マウスによる簡単操作、65,536色中16色の多彩な表現、クリエイティブマインドに応えるウィンドウ対応のペイントツールです。同時に複数のウィンドウを開いて編集でき、各ウィンドウ間のデータ交換も行えます。

●定評のGUI対応ウィンドウワープロ

EGWord SX-68K

CZ-271BWD　標準価格59,800円(税別)

キャラクタベースのワープロを超えたGUIによる、手軽なDTPソフトとしても優れた表現力を発揮。定評ある日本語入力方式によるインライン入力、各種グラフィックデータやテキストデータの貼り込みができます。

4MB | ver.2.0 | HD 5MB

●グラフィック感覚の楽譜入力をサポート

MUSIC SX-68K

CZ-274MWD　標準価格38,000円(税別)

MIDI、FM、ADPCMに対応した楽譜ワープロ&作曲演奏ソフト。自由なレイアウトで、グラフィックを描くように楽譜入力。全パートの同時入力・編集、自動伴奏機能、多彩なプリンタ対応で美しい印刷も行えます。

4MB | ver.3.0

●マルチタスク機能をはじめ通信環境がさらに充実

Communication SX-68K

CZ-272CWD　標準価格19,800円(税別)

通信環境をさらに高めたウィンドウ対応の通信ソフト。マルチタスク機能により他のアプリケーションを実行中でも簡単に通信が可能。自動ログイン機能やプログラム機能など、豊富な機能をサポートしています。

2MB | ver.1.1

開発支援ツール

●X68030/X68000対応開発ツール

C COMPILER PRO-68K ver2.1 NEW KIT

CZ-295LSD　標準価格44,800円(税別)

C compiler PRO-68KのX68030/X68000対応版。従来からの機能に加えて、Human68k ver.3.0、ASK68K ver.3.0にも対応。新たにGPIBライブラリ、MC68882対応フロートライブラリを付属しています。

2MB

●SX-WINDOWソフト開発支援ツール

SX-WINDOW 開発キット Workroom SX-68K

CZ-288LWD　標準価格39,800円(税別)

SX-WINDOW用のソフトウェア開発に必要なツールや33種類のサンプルプログラムを装備。プログラムの編集、リソースの作成、コンパイル、デバッグといった一連の作業がきわめて効率よく実行できます。

※ご使用に当ってはC compiler PRO-68K ver.2.1が必要です。

4MB | ver.2.0

●SX-WINDOW開発キットのサポートツール

開発キット用ツール集

CZ-289TWD　標準価格12,800円(税別)

「SX-WINDOW開発キット」をさらに使いやすくするためのサポートツール集。SXコールの簡易リファレンスを収めたインサイドSX、イベントハンドラ、ヒープビューアなど11種類のツールが用意されています。

4MB | ver.2.0

4MB | ver.3.0 | HD 10MB の表示は、メインメモリ4MB以上、SX-WINDOW ver.3.0以上、10MB以上の空きのあるハードディスクが必要であることを示しています。●EGWordは株式会社エルゴソフトの登録商標です。

●お問い合わせは … シャープ㈱電子機器事業本部(液映)システム機器推進プロジェクトチーム　〒162 東京都新宿区市谷八幡町8番地　☎(03)3260-1161(大代表)へ　シャープ株式会社

資料請求券 CZソフト Oh!X 9係

特報!

バーチャファイター2
セガラリー・チャンピオンシップ

好評発売中!!
毎月8日発売

バーチャコップ／機動戦士ガンダム
ガーディアン・ヒーローズ
真・女神転生 デビルサマナー
アリーナ

特集：夏の3大RPGにくびったけ！

シャイニング・ウィズダム
リグロードサーガ
魔法騎士レイアース

［AM2研EXPRESS NEO］
緊急登場「VF2.1」と
「VF3」最新速報!!

▼COMING SOON SOFT
発売目前!
セガサターンソフトを大紹介!
ワールドアドバンスド大戦略
シムシティ2000
ストリートファイター リアルバトル オン フィルム
Dの食卓／卒業II ネオ・ジェネレーション
プロサッカークラブをつくろう
ウィニングポストEX／メタルファイター♥MIKU
その他、全24本一挙紹介！

▼SEGA SATURN SOFT COMPLETE GUIDE
発売後のセガサターンソフトを徹底攻略!
実況パワフルプロ野球'95 開幕版
麻雀海岸物語～麻雀狂時代セクシーアイドル編～
ゆみみみっくすREMIX
クロックワークナイト
～ペパルーチョの大冒険・下巻～

CDサイズ特別付録（小冊子）永久保存版裏技辞典
SEGASATURN
SECRET TECHNIC FILE
エンディングまでイカせる裏技も満載!

■定価は税込みです ■お近くの書店でお求め下さい
ソフトバンク株式会社／出版事業部 販売局 TEL.03-5642-8100

ファン待望、初の原画集ついに登場!!

アイドル雀士 スーチーパイ 原画&設定資料集

株式会社ジャレコ 監修

ゲームセンターのみならず次世代ゲーム機でも人気沸騰の「アイドル雀士 スーチーパイ」。ガルフォースやガンスミスキャッツなどで知られる園田健一氏の原画はもちろん、カラーCG、スーチーパイの歴史、スーチーパイ図鑑に加え、声優インタビュー、開発インタビュー、新たに作曲されたテーマ曲の譜面を収録するなど、ファンにはたまらない盛りだくさんな作りになっています。

A4判
定価1,900円

©1995 JALECO LTD.

好評発売中

スーパーリアル麻雀PV 原画&設定資料集

スーパーリアル麻雀シリーズ最新版PVの未公開設定資料満載。動画枚数1000枚突破のアニメーションシーンもバッチリ完全収録。おなじみのピンナップ付録に加え、巻末に“飛び出すPVポップアップ”が付いています。

A4判・定価2,000円

スーパーリアル麻雀PII&PIII ファンブック

A4判・定価2,000円

スーパーリアル麻雀PIV 原画&設定資料集

A4判・定価2,000円

LUNAR I・II 公式設定資料集

メガCD史上最高傑作RPGとの呼び声の高いLUNARシリーズの公式設定資料集。キャラクターデザインを担当した窪岡俊之氏の描き下ろしイラストや佐藤肇氏による世界設定イラストなど、貴重な資料をあますところなく掲載。

A4判・定価2,800円

闘神都市II 原画&設定資料集

アリスソフト 監修

大ヒット中のパソコンRPG超大作「闘神都市II」の原画&設定資料集。アリスソフトの貴重で美麗な開発資料をページの許す限りてんこ盛り。さらに、全マップからサブイベントまで徹底攻略。特製ピンナップつき。

A4判・定価2,500円

●定価は税込みです　●お近くの書店でお求めください

ソフトバンク株式会社／出版事業部
販売局 TEL：03-5642-8101
SOFT BANK

SX-WINDOW ver.3.1 開発キット

著
◆
吉沢正敏
牛島健雄
西田文彦
小浜 純

B5変形判580ページ
定価5,800円
5"FD 1枚＋CD-ROM 1枚付

◆
本書の内容
◆

本書は、シャープ提供の開発環境「Workroom SX-68K」と、
『追補版SX-WINDOWプログラミング』などで提供されたフリーソフトによる
開発環境を統合し、最新のSX-WINDOW ver.3.1の機能を利用した
アプリケーション開発環境を提供するものです。
添付FDには本書の著者たちが推奨する開発環境とCD-ROMドライバが、
添付CD-ROMには200本弱のSX-WINDOW対応フリーソフトを収録しています。
また、巻末にはver.3.1までのすべてのSXコールリファレンスをまとめてあります。

NetBSD/X68k

NetBSD/X68k委員会◆著

5"FD 1枚＋CD-ROM 1枚付き

SOFT BANK ソフトバンク株式会社出版事業部 〒103 東京都中央区日本橋浜町3-42-3 TEL03-5642-8101

[第52回]
あるホームレスのこと

江口　響子

響子inCGわ〜るど

毎週金曜日は，八王子にある大学に出かける日だ。なにしろ遠い。通勤時間は約２時間である。たいてい，７時39分新宿発の通勤快速に乗る。この電車を逃してしまうと次は７時53分で，授業開始にはぎりぎりだ。その次は８時６分になってしまい，完全に遅刻である。

ここまで書くと，なんだか西村京太郎の鉄道推理小説めいてくるが，もちろん話はそうではない。時間にこだわったのは，あるひとりのホームレスのことを書きたかったからだ。

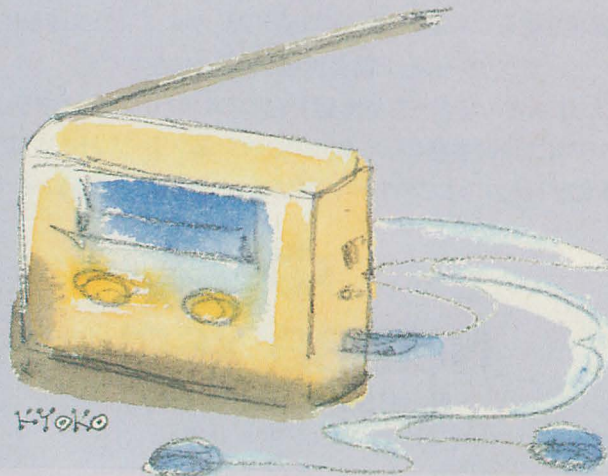
朝の時間は貴重だ。JR新宿駅から京王新線新宿駅まで，自分なりの近道をして歩く。誰でもそうしているのだろうが，改札口から改札口をできるだけ対角に結ぶように動く。階段や通路のどのルートをとるか，けものがけもの道を進むように決まっている。彼はその通り道にいた。

京王線の改札口の前，階段の脇１メートル50センチ四方が，彼のテリトリーらしい。白髪まじりの日焼けした風貌は，どことなくニュースキャスターの筑紫哲也氏のようだ。積み上げた本を椅子がわりにして，壁に向かって座っている。両脇に黄ばんだ雑誌や本がうず高く積まれていて，ダンボールは見当たらない。

私が初めて彼を見たとき，うつむいてなにかをじっと読んでいた。そばに寄って覗き込みたかった。が，失礼な気がしたし急いでもいたので，横目で窺いながらゆっくりと通り過ぎた。どうやら薄い洋書のようだ。ぼうぼうに伸びた髪の間から，ヘッドフォンのコードが伸びている。コードの先に携帯ラジオがあった。

そうなのだ。彼は，ラジオの語学講座を聞いていたのである。少し注意を払って見てみると，洋書に見えたのはテキストだったのだ。この時間だとNHKなら確かハングル語かフランス語のはずだが……。

７時39分発に間に合うときは，彼は必ずテキストを開いて座っていた。たまに遅れて７時53分発の電車になってしまったとき，いつも彼はいなかった。時間に正確な人なのである。

新宿駅西口地下街は，ホームレスの人たちがとても多い。男も女もいる。たいていはダンボールの中にころがって寝ている。地下街は外気とほどよく遮断されているので，冬は暖かい。が，その分夏場は暑い。７月でもダンボールで囲っている

のは，雨風を防ぐというより道ゆく人の好奇の視線をさえぎるためであろう。積み重なったダンボールの家からは，かすかに小便の臭いさえする。通勤の人や買い物にきた人は，すえた臭気に顔をしかめながら，さっさと通り過ぎる。

そんなホームレスの人たちの中，自分を堂々と人目にさらし語学のテキストを開く筑紫哲也氏は，なかなかに目立っていた。1メートル50センチ四方の書斎を覆う見えない壁は，ダンボールよりはずっと丈夫なのだろう。

現在，引っ越したため金曜日の朝のけもの道は変わってしまった。もう新宿は通らない。あの筑紫哲也氏と出会うことはないと思う。残念だが。

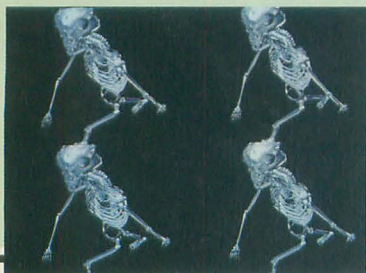
今回のCGデータ

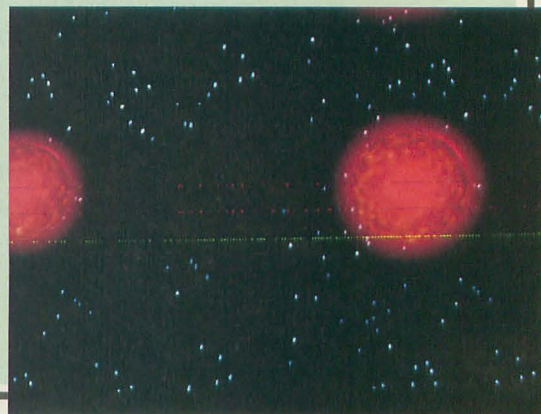
1280×1024ピクセル
1670万色フルカラーを4×5ポジで出力
作成手順
背景はMATIERで作成。スケッチ画像を裏画面に置いて表画面にコピー，ネガティブ，カラーレリーフ，指先ツールなどで画像処理。手前の立方体はサイクロンで作成。マッピングとCSG。

[特集]

Animation Now!

DoGA CGAシステムとXL/Imageで作成されたアニメーション映像の数々（映像協力：森山知己氏）。これらがハードディスクからノンストップで再生可能となった。65536色映像なのでHANIMを使用する場合よりクオリティアップも可能です。ま，こういうものは動いている画面でないとわからないとは思いますが，雰囲気だけでも味わってください。

サターンおよび3DOのアニメーションデータ再生を行った例。ちなみにサターンは最新のシステムではまたデータ形式が変わってしまったようだ。

AMIおよびHDANIM形式でのデータの配列の例。X68000のG-RAM構造を基本としている。

X68030のバスマスタモードでG-RAMにデータを読み込んだときに発生するノイズ。原因は不明だ

THE USER'S WORKS in TAKERU

●プリンセスクロワッサン/GUARDIAN・RS/CLISS/情け無用Fire!&2●

個人レベルの創作活動が比較的活発なX68000。現在までいろいろな同人ソフトが発表されてきました。

しかし，それらのソフトの情報を知る手段は，かなり限られています。それに同人ソフトは，個人どうしの取引ですからいろいろと煩わしい手間がかかります。興味はあるけど郵送などの手間をかけてまで……と購入を躊躇していた人もTAKERUを利用すれば，簡単に購入することができます。それに，値段もかなり安く，ソフト1本あたり500円～2,3千円の範囲で購入できます。郵送費を含めても，結構割安なお値段で購入できるのもTAKERUを利用する魅力のひとつでしょう。

ただし，同人ソフトというものは，技術力的にばらつきがあるためハズレを引かないようにするためにも，それなりの情報収集が必要になります。もちろん，TAKERUの検索段階でゲーム内容の説明と画面写真も表示されます。しかし，それだけでは十分な判断材料となりえるとはいえません。

いくら値段が安いからといっても，自分の好みでないゲームを購入してしまうという失敗をしたくありませんからね。TAKERUを利用できる環境にいる人たちは，ソフトを購入する際に参考にしてください。

プリンセスクロワッサン

●人面狐工房/X68000

月の裏側にある老舗の温泉旅館の一人娘(クロワッサン)が，親子ゲンカの末に地球に家出。そして，家出先でなぜかクロワッサンの面倒を見ることになった地元高校生竹本光(主人公)。そして，母親が家出したクロワッサンを連れ戻そうとしたところから，騒動が始まります。チミモーリョーをバラまきながら逃げるクロワッサンを，高額報酬につられて家に連れ戻すことを了解してしまった竹本光の未来やいかに?!

以上がこのゲームの大まかな(?)ストーリーです。ゲーム画面も，全面ギャグをちりばめたものとなっています。敵は人面取り込みキャラクターやマッチョな兄貴，そしてご丁寧に将棋の“金”と“玉”が並んで登場する始末。特に1面のボスが登場したときには，思わずマウスの右ボタンを押したかどうか確認してしまうほど笑えます。

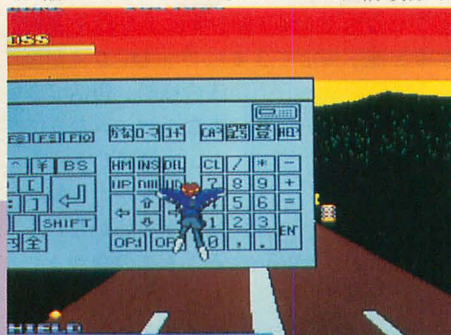
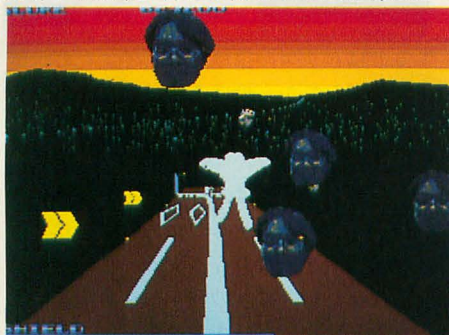
このようにギャグがポイントの「プリンセスクロワッサン」は，ダメージ制の3Dシューティングゲームです。群がる敵をガンガン撃ち落とし，最後に現れるボスを倒せばステージクリア。ステージは，両脇に民家が立つ国道らしきステージや，ビル街の間をすり抜けるようなステージ，そして通路内を突き進んでいくステージなど，全部で6ステージ。処理速度もそこそこで，快適にゲームを遊ぶことができるでしょう(10MHzでは多少処理落ちが目立つ)。

また，設定はナンパでもゲームシステム自体は，ポリゴン描画にビットマップの拡大縮小をバリバリに使っている結構硬派な作りです。そのほかにも，地形を間引いて処理を軽くしたり，マシンの速度に応じたゲーム速度にできるという環境設定もできます。しかも，ゲーム中に随時変更が可能という親切設計です。

ただし，敵キャラクターのパターンが少なく，難易度設定もかなり低いため，何度もプレイしてゲームを極めてやろうという気にならないのが残念なところです。逆にいえば，最近のゲームは難しすぎてやってられない，なんて人にはお勧めかもしれません(もちろんギャグモノにアレルギーのない人にかぎられますが)。

価格：1,200円(税込)

〈総合評価〉

技術力：★★★★★★★☆☆☆
お笑い：★★★★★☆☆☆☆☆
音楽：★★★★☆☆☆☆☆☆
お買い得度：★★★★★★★☆☆☆

GUARDIAN・RS

●TRAP・零/X68000

とにかく回転，拡大縮小したい人(?)にお勧めなのがこのゲーム「GUARDIAN・RS」だ。画面中央に配置された自機を操作(左右方向で旋回，上下方向で上昇，下降，Aボタンで加速，Bボタンで武器連射)し，プレイヤーは，次々と襲ってくる敵を撃破しなくてはならない。

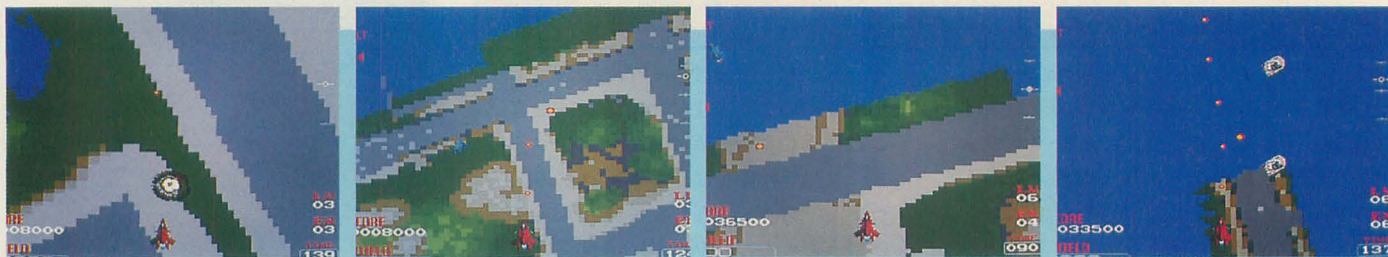
そして，最終的にシールドがなくなるか，一定時間以内に決められた敵機数を破壊できないとゲームオーバー。さしずめ，高度コントロールのできる簡易「タイムパイロット」といったゲーム内容である。しかし，遊んでいてなんかつまらない。これは，マニュアルにある内容紹介を読むとわかる。「BG高速回転拡大縮小システム搭載。BGを2枚重ねることにより，疑似的な64×64画素を表示，および毎秒40枚以上(16MHz時)の高速展開を可能とし，迫力とスピード感を演出した3Dフライトアクションゲームです」

つまり，あくまでもメインとなっているのは，回転拡大縮小システムなのである。最初のうちは「すっげえかもしれない」と思いつつ遊べるのだが，ゲーム性はほとんど皆無。敵を倒しても倒しても，レベルが上がるだけで，あっという間に飽きてしまう可能性がある。背景のバリエーションを増やしたり，ボス格のキャラクターを用意したり，それぞれの面でいろいろなミッションをやらせるなどして，ゲームとしての味つけをしっかりしてほしかった。結局，

せっかくのシステムが，ゲームに活かされていないのが残念である。

価格：800円(税込)

〈総合評価〉

技術力：★★★★★★☆☆☆☆
爽快感：★★★★★☆☆☆☆☆
音楽：なし
お買い得度：★★★★☆☆☆☆☆☆

CLISS

●ふぇにっくす/X68000

「CLISS」は，魔法の風船を持ったふよふよ漂うクリスを上下左右に操り，最終的に雲の国へ導いてあげることが目的のアクションゲームだ。

基本的に一定速度でクリスは上昇していき，一定高度まで無事たどり着けばステージクリアとなる。ゲームでは，空飛ぶクリスが珍しいのか，いろいろな敵がちょっかいをかけてくる。もちろん，それらの敵に触れてしまうと風船が割れて，クリスは地上に向かって落下し始める。普通だったら敵に触れるとその場で1ミスとなるが，このゲームでは，風船を失うと判断し，ある一定位置まで落下すると自動的に復活するようになっている(落下中に上方向を連射すると速く復活できる)。う～む，さすが魔法の風船。復活できるならゲームなんてちょ～簡単じゃんなどと思うが，実際にはそうもいかない。そこそこの高さまで上がっていると簡単にミスとはならないが，当たり判定が大きいため，ぼんやりしていると落下中にも敵の攻撃を受け，あっという間に高度を失ってしまうのだ。

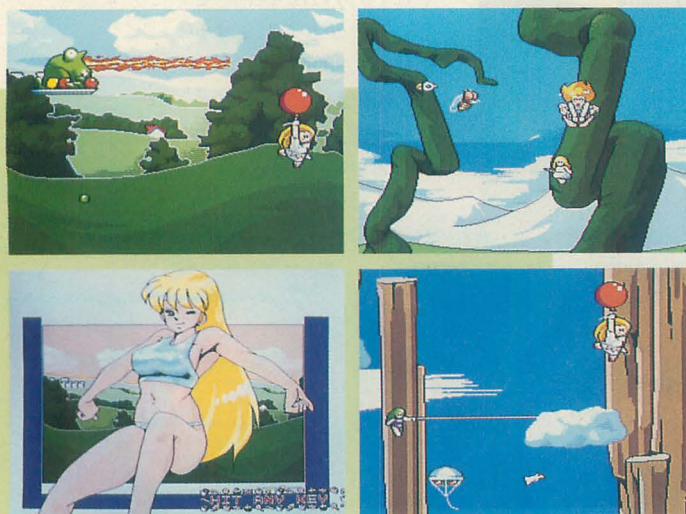
もちろん反撃の手段もあり，敵を撃退するのはクリスの投げるお菓子。ただ，ボタンを押してから投げるまでのタイムラグがあるので，ある程度敵の動きを予測して投げる必要がある。

ゲームデザインは，ちょっとだけ古くさいが，とにかくクリスがかあいい。ほかの敵キャラクターたちも多少統一性がないかもしれないが，動きのパターンが増えればいうことなし，というところだ。

あと，ステージクリア後に表示されるサービスCG(ちょっとH)はいらなかったかも。

価格：500円(税込)

〈総合評価〉

技術力：★★★★☆☆☆☆☆☆
クリス：★★★★★★☆☆☆☆
音楽：★★★★★☆☆☆☆☆
お買い得度：★★★★★★☆☆☆☆

情け無用Fire!&2

●YAMACO Software/X68000

1対1で繰り広げられる熱い対戦タンクゲームが，この「情け無用Fire!」「情け無用Fire! 2」です(以下F1，F2とします)。

基本ルールは，とにかくフィールド内をかけ巡って撃ちまくり，相手を破壊したプレイヤーが勝ちというもの。

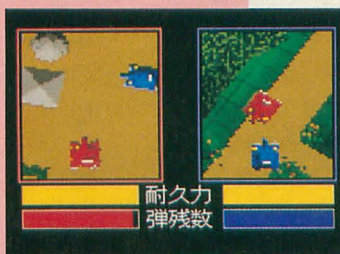
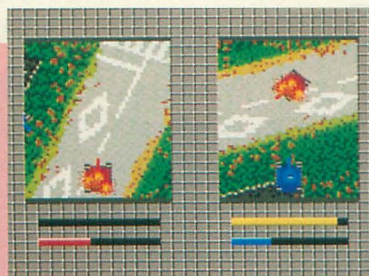
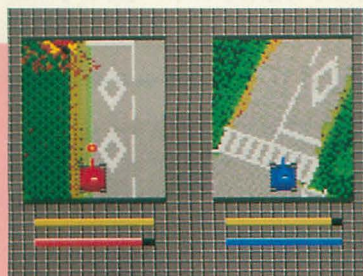
戦車の操作はF1，F2ともにほぼ同じ。左右方向で旋回，上下方向で前進後退，トリガAで弾の発射となっています(F2では，トリガBを押しながらスティックを倒すとカニ歩きもできるようになっています)。戦車の耐久力は，前面が高く，側面，背面は低く設定されています。つまり，できるだけ相手の背後をとることが，勝つための条件といえます。

そして，対戦するフィールドは，F1が平面マップ，F2が障害物などの地形のあるマップになっています。平面マップのF1は，ひたすらケツの取り合いに終始してしまいがちですが，F2では障害物があるため，地形を利用した戦略が必要になり，ゲーム性が上がっています。そして，ゲーム中にはランダムにアイテムが登場します。F1は，取ったアイテムによって戦車の性能そのものが変化するようになっていて，F2は兵器の性能アップ(4種類)にレーダー，カモフラージュ，相手の攻撃を封印するというようなものが用意されています。

結構ありがちなゲームですが，やはり対戦ゲームは燃えます。ゲームを遊ぶためには1本のジョイスティックと1人の人間が必要なのは残念ですが(F1では1人プレイモードもあるが，コンピュータ戦車が卑怯で遊んでいても楽しくない)。

価格(F1)：500円(税込)
価格(F2)：1,300円(税込)

〈総合評価〉

技術力：★★★★★★☆☆☆☆
対戦：★★★★★☆☆☆☆☆
音楽：なし
お買い得度：★★★★☆☆☆☆☆☆

SOFTWARE INFORMATION

久しぶりの新情報。計測技研から「シャーペンワープロパックver.2.0」の発売が決定しました。

基本編集機能，印刷機能と細かいところでさらに機能アップがされているようです。

発売が楽しみなソフトですね。

このソフトについては，サンプル(もしくは製品版)が届きしだい，詳しくレポートを行う予定です。

そのほかの「EXITINGみるく」以外は，特に動きを見せていません。状況が状況なだけに，各メーカーの皆さんには，がんばっていただきたいものです。

さて，今月から始まった「THE USER'S WORKS in TAKERU」では，新旧こだわらず，TAKERUに収録されている同人ソフトを紹介していきます。また，いままでどおりの「THE USER'S WORKS」も募集しています。読者の皆さんが制作した，パワー溢れる作品をお待ちしています。

新作情報

★EXCITINGみるく　TAKERU　10/未
X68000用　5″/3.5″2HD版　1,500円(税込)
★X CASE　Beシステム
X68000用　5″2HD版　19,800円(税込)
★Traum　象スタジオ
X68000用　5″2HD版　価格未定
★麻雀悟空・天竺への道　シャノアール
X68000用　5″2HD版　9,800円(税別)
★地球防衛MIRACLE FORCE　カスタム
X68000用　5″2HD版　価格未定
★プリンセスメーカー　ニュー
X68000用　5″2HD版　14,800円(税別)

甦れ，黄金龍の伝説

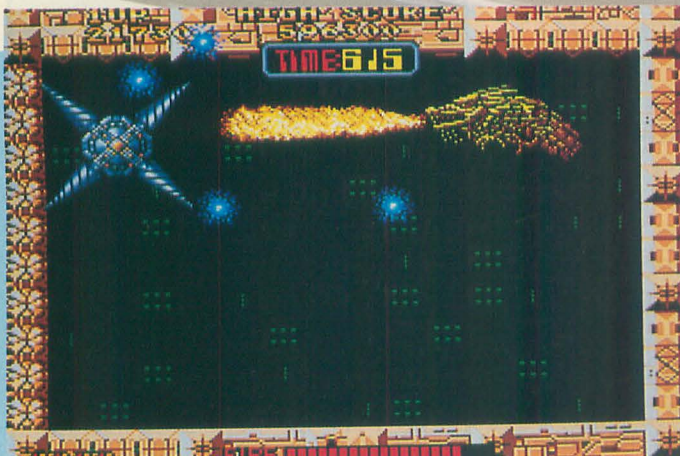
Yaegaki Nachi 八重垣 那智

ゲームの懐に飛び込んで，初めて面白さがわかる作品がある
少し遊んだだけでは，見すごしていまいそうな仕掛けを分析しつつ
そんなゲームの面白さを探ってみよう

●サイバリオン

シャープ　☎03(3260)1161

最近，またチョロチョロとシューティングがゲームセンターにお目見えしてきた。しかし独創的なヤツは，やっぱり皆無。どのゲームもどこかで見たようなルールのヤツばっかりだ。安心して遊べるのはいいのだが，いつでも諦められるような軽さが感じられてしまい，いまいちのめり込めない。つまり，プレイヤーのリピートに対する欲求を引き出し切れないという問題を抱えているように見えてしまうのである。

そういういまどきのシューティングゲームを見ていると，思い出すゲームがいくつかある。特定のプレイヤーしか見ていない，アクの強いゲーム。制作者の思想のもとに，多くのプレイヤーのウケを切り捨て，ついてくる者だけに特権的な快楽を提供したゲーム。これらのゲームは，たとえその外見の印象がよくなくても，ゲームに二歩も三歩も踏み込んで戻れなくなったとき，初めてその真の姿を見せてくれる奥ゆかしさが特徴だ。いまは絶滅してしまった感のある，こういったゲーム(プレイヤー層の狭さから三日月型ゲームと呼ぶのがふさわしいだろう)は，ゲームセンターではなく改めて家庭の中で集中してプレイすれば，比較的容易にその境地に達することができると考えられる。

そこで今回は，その典型中の典型である「サイバリオン」のX68000移植版をターゲットとして，ゲームそのもの以外に三日月型ゲームの楽しみ方から，その限界まで一般論を含めてすべてをここで再評価することを試みる。

黄金龍の過去

まずは「サイバリオン」の生い立ちを紹介しよう。「サイバリオン」の存在が知られるようになったのは1988年の6月，実に7年も前のことだ。実際に各地に出回るのは10月頃なのであるが，当時あの「バブルボブル」や「ハーレーズ・コメット」の企画者であるタイトーのゲームデザイナー「MTJ氏」の最新作として，巷のマニアには疾風のように情報が駆け巡っていた。「とにかく普通のゲームとは違うらしい」という，極めて曖昧かつ期待を含んだ噂は，まだ見ぬゲームに対する，無条件な称賛の言葉でしかなかった。

当時ロケテストで「サイバリオン」に遭遇した私には，その特異さが印象深い記憶として残っている。トラックボールによる操作系，そして高精細ディスプレイによる緻密なグラフィック。もちろん，こうした見てくれだけでなく，内容もより特異だった。状況や展開というゲーム内容自体がプレイごとに変わるゲームシステム。そこで展開されるサブストーリーは，分岐を繰り返し100以上もの結末が待っているというマルチエンディングの採用。どれもこれも既存のゲームには存在しない概念によって組み立てられており，それは当時知りうるゲームの中で，最も特殊で類を見ないものであった。

跳ねまわるボス。隅に追いつめるが吉

しかしそれゆえか，その三日月の芸術的なまでの細さというものには，当時の私は気づくことができなかった。実際は，新しいことの多さに惑わされていただけなのかもしれない。

龍の姿を知るべし

ゲーム外部の独創的なシステムは，前述のとおりだ。実は「サイバリオン」は基本的なゲームシステムも，独創的なものを採用している。軽く説明しよう。

まず自機が龍である。龍は多関節でできており，胴体や尾はプレイヤーの操作する頭の動きをトレースする。当たり判定はその長大な全身にあり，その大きさどおり8回の敵の攻撃に耐える仕組みになっている。残り耐久力はプレイヤー(龍)自身の色の変化で表現してあり，瀕死の場合は警告も出るので，自機がどんな状況であるか比較的捉えやすくなっている。

そして自機の攻撃方法は一定距離に伸びる炎。攻防一体で，敵の攻撃を弾き返すことができるのが特徴だ。ただし威力を示すメーターがついており，使用中はそれが減っていき，それに対応して炎が縮む仕組みになっている。メーターは非攻撃時の自機の移動速度に対応して回復するので，緩急を使い分けたメリハリのある操作が要求されることになる。すでにこの時点で，トラックボールを使うことを考えると，相当に忙しいゲームであることがわかるだろう。

ゲームの目的は敵と戦いながら折れ曲がった通路を抜け，その終着点で待っているボスを倒すことにある。もちろんこの通路はステージごとに自動作成されて毎回構造が異なる。さらに，終着点はたどり着くまでどんなボスが待っているのかもわからない。しかも，ボスを含めたあらゆる敵はプレイヤーの腕を判断し，攻撃方法からその挙動までをも変えてくるので，より厄介で

自分のミスは護衛機を失うことになるぞ

ミスしても護衛機が増えることもある

複雑な事態を呈している。

もうここでわかったとは思うが，「サイバリオン」にデータ記憶の学習による攻略という概念はない。ゲームの場面に合わせてプレイヤーは，臨機応変に対応しなくてはならないのだ。これが「サイバリオン」最大の特徴である。ただし例外というか，マップとボスが固定された「練習」モードというのが初心者向けに用意されている。ここでは通常見られるゲームのような，記憶と学習の関係が成立するが，ストーリー展開といった比較的重要な部分が排除されているためにあくまでもオマケ，形式的な性格が強い。そのためここでは，練習モードについてあまり重きを置かないことにする。

龍が誘う輝き

こういった，簡単にはゲーム内容すら語り尽くせない「サイバリオン」であるが，その複雑さから，ゲーム自体に関していささか誤解されているようなところがある。これについて触れておこう。

この「サイバリオン」におけるゲーム中の戦闘は，あまり重要な要素ではない。むしろ正確にかつ安全に早く通路を次々と突破していけるような心掛けをしたほうが，結果としてはよいことになる。理論上一部の障害物を除いて，倒す必要のある敵はボスだけなのであるから，変にアイテムや得点に執心して無駄な戦闘をするべきではないといえる。いわゆる初心者にありがちな，牛歩のように進みながら出てくる敵をいちいち相手にするというプレイスタイルでは，このゲームは少しも面白くない。

確かにアクションゲームなので，攻撃的要素が前面に押し出されてはいる。しかし，それはあくまでも形だけであり，このゲームが比較的レースゲームに近い様相を呈していることは押さえておくべきことであろう。滑らかに自機を操り，止まらずに美しく障害物を次々に突破していくところに，このゲームのプレイヤーは美を意識するべきなのだ。「サイバリオン」をこの点で誤解し，ノリの悪い戦闘で目的がわかりにくいアクションゲームだ，というように捉えてしまうとこのゲームの懐には入り込めない。誤解しているような人は，いまからプレイしなおしてでも，そういった認識を改めてもらいたいものである。

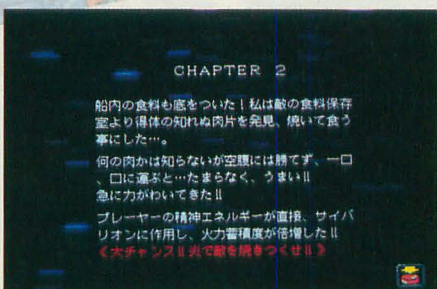
またここではX68000版独自のお約束についても書いておかねばならないだろう。X68000版がオリジナルと大きく異なる点は，大きく挙げて2点ある。操作系とグラフィックだ。どちらも外見的部分だけに，オリジナルと異なることは，移植の印象を大きく左右しかねない部分であることに注目したい。

鏡写しの龍

操作系の違いは，基本的にジョイスティックでプレイすることになるということが大きなウェイトを占める。確かにマウスやトラックボールにも対応してはいるのだが，あまりこのゲームのためだけにトラックボールを用意するのも気が引けるものがある（ソフトよりも価格が高いのでなおさらその感は強い）。また，トラックボールは業務用のものと比べると耐久性にも疑問がある。細かいことを気にするぐらいならジョイスティックで十分だろうと，割り切るほうがよさそうだ。

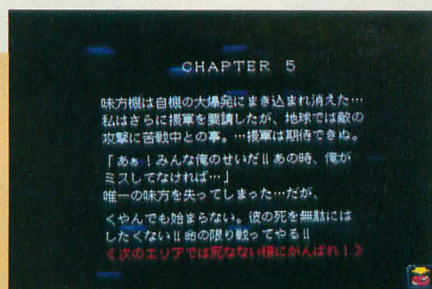
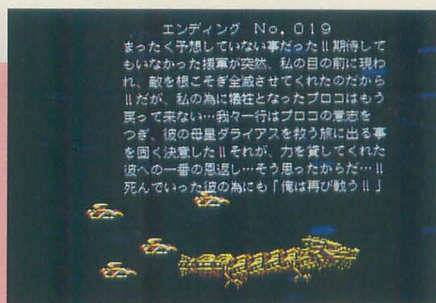
実際にオリジナルをプレイしていると気づくかもしれないが，トラックボールで操作しても8方向にしかセンスされず，むしろ移動速度の使い分けのために利用されているといっていい。そういうことを考えると，ジョイスティックでの操作でもあまりゲーム性を下げずにプレイできるといえる。ちなみに本体付属のマウス・トラックボールのトラックボールは弾みで回らないため，ちょっと試してみればすぐに役に立たないことがわかる。さすがにこれでプレイする人はいないだろう。

もうひとつの違いであるグラフィックである。これはゲーム性にはほとんど影響がないものではあるが，あえて異論を唱えておきたい。オリジナルのグラフィックは，特殊な画面周波数を用いて，通常のビデオゲームには珍しい高精細グラフィックを実現しており，これは外見的な「サイバリオン」の最大の魅力であったと私は確信している。

しかし，実際のX68000版では通常の解像度に画面解像度を変更してあり，それに応じてキャラクターには修正がかけられている。そこいらの家庭用ゲーム機に移植した

よい展開の少ない「焼肉」パターン

「急げ」などの指示は守らないと悪い展開に

力をあわせて攻撃。こいつが最後の敵だ

エンディング No. 019
まったく予想していない事だった‼期待してもいなかった援軍が突然、私の目の前に現われ、敵を根こそぎ全滅させてくれたのだから‼だが、私の為に犠牲となったプロコはもう戻って来ない…我々一行はプロコの意志をつぎ、彼の母星ダライアスを救う旅に出る事を固く決意した‼それが、力を貸してくれた彼への一番の恩返し…そう思ったからだ…‼死んでいった彼の為にも「俺は再び戦う‼」

ありがちだが一応ハッピーエンドである

のではないのだから，グラフィックをコンバートする手間を経るぐらいなら，ひとつX68000ならではという部分を見せてもらいたかったと，いまでも本気で思っている。漢字のメッセージなどは画面モードを切り替えて表示するなど，あまりスマートな処理に見えないあたりが，さらに印象を悪くしているように感じられてならない。トラックボールの問題と異なり，物理的な要因ではないわけだから，簡単に納得し難い部分であると主張し続けたいと思う。

長くなってしまったが，ここまでで「サイバリオン」というゲームの特殊性と，そのX68000版についての知識は備わったと思われるので，次から「サイバリオン」に代表される三日月型ゲーム自体の分析に入りたいと思う。

深き龍の瞳

まず一般論として，繰り返しプレイによるゲームの奥深さを演出するには，どういう要素が必要なのか考えてみよう。まず複数回プレイが必要になる全体構成というのが挙げられる。つまりキャラクターやルート選択，ランダムイベントといった，一度のプレイで全体の一部しか経験できないという択一的な構造のことである。戻ることのできない場所で選択を行わせることで，冒頭から再プレイしない限り，残されたルートに対する情報や楽しみを手に入れることはできない。童話の舌切り雀で，お土産のつづらを選ぶような状況が，ゲームによっては矢継ぎばやにプレイヤーに対し迫ってくるのだ。

分岐が多ければ多いほど，また深ければ深いほど，それらをすべて制覇するための必要プレイ回数は指数関数的に増大していく。このためこれらを網羅するための未知の領域への興味がリピートプレイを生み出すことになる。すべてのルートを制覇するために，すべてのエンディングを見るために，そしてすべての演出を体験するために，プレイヤーは再びスタートボタンを押すのである。気力(や財力)の続く限りか，あるいはその欲望が満たされるまで挑戦が続いていくのである。

さらに奥深さを構成する要素は，もうひとつある。それは果てしない，極めて高い難度だ。ただこれは，単純にプレイヤーを拒絶する一様な難しさというものではなく，プレイヤーの腕を判断し，それに追従して青天井に難しくなるような比較的複雑な難易度システムのことである。

また，プレイヤーにとって本来のゲーム内容から要求されている動作を遥かに上回る条件のボーナスシステムが，過度に用意されているようなことも，間接的な難度として考えていいだろう。これらの要素が豊富にあることで，プレイヤーはどのプレイにおいても必ずやり残したことや，プレイに対する不満が生まれてしまい，たとえゲームがクリアできたとしても，より高い目標を(それこそどこまでも)示され続けることになる。ゲームに対し，どこで満足するかは個人の自由だが，目標が示され続ける限り，それはリピートプレイの立派な動機となる。ここではプレイヤーの向上心を刺激するという方法で，その欲求を生み出しているのだ。

龍による掌握

では「サイバリオン」はこれらの条件をどのように実現しているのか検証してみることにしよう。まず分岐による選択性についてだが，これは主にサブストーリーにおいて顕著である。ストーリーの分岐は乱数によるものもあるが，基本的にはゲーム中の行動を場面に応じて分岐させている。選んでいるという意識は低いものの，最終的に103種類まで分岐するため，ずいぶん大きな仕掛けに思えてくる。つまり単純に考えても103回のプレイが必要であるからだ。途中のステージで経由したストーリーについては，全体に対しての位置づけがわかりにくいが，エンディングのときだけストーリーには番号がついており，これが征服欲を刺激するようになっている。

しかし，特定の番号のエンディングに到達する方法が不明なので，仲間内で自分が見たエンディングの情報を集め，ストーリーの全貌に迫るような試みをしていた人たちもいたようだ。個々のストーリーの内容は陳腐なのかもしれないが，こういった全体の規模の魅力では，ほかに類を見ないも

無敵は時間短縮のチャンス。一気に進め

反射弾は散らばる前に炎で食いとめろ

新兵器らしいが，実は役に立たない砲台

やはり勝負は己の炎でケリをつけろ

のになっている。

もう一方の難易度に関するフィーチャーだが,「サイバリオン」ではこちらの仕掛けも豊富になっている。難度はノーミスボーナスや得点アイテムの連続回収といったものやステージの進行により，徐々に上がっていくようになっている。これによりボスの攻撃や動きが変化するだけでなく，性能の高い雑魚が出現したりするようになるので油断がならない。もちろん自動生成される通路も作為的に狭く作られるようになるので，初心者と上級者のゲーム画面を比べると，そのあまりの違いの多さに驚かされる。

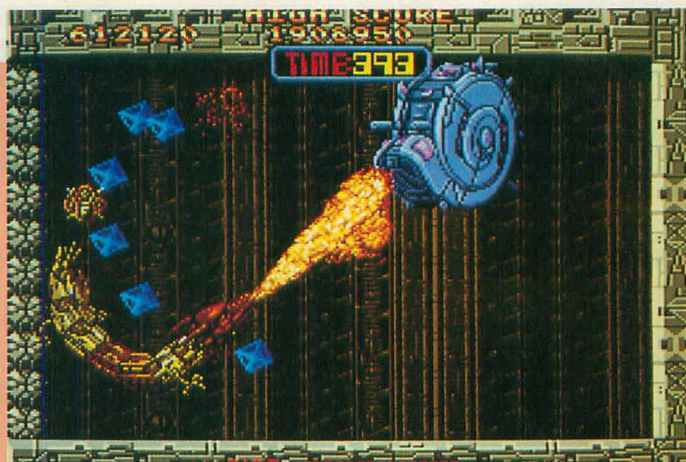
これがさらに推し進められ，最終ステージで一定以上のスコアを出しているときに「隠し」ボス（しかも2種類いる)が登場するという，このゲーム最大のイベントに到達できる。しかも条件は単一ではなく二段構えになっており，より強い隠しボス(もちろんこれも2種類)なるものまで用意されている。ここにいたってはどれだけの人が，存在すら知っていたかどうか疑問に感じるほど果てしなく高度なレベルになっている。このように難度に関しては青天井の上昇を見せるため，常にプレイヤーは腕に応じた攻撃にさらされる。常に新鮮な感覚でプレイすることができるといえるかもしれない。

こうしたシステムをもつ「サイバリオン」を楽しむ方法としては，到達したエンディングの記録とスコアアタックに尽きる。特に隠しボスの出現条件を左右しているノーミスボーナスを取れるようにするのが，最初のステップになるだろう。とにかくプレイを重ね，このゲームに詰め込まれた仕掛けをできる限り多く体験することが，このゲームをプレイする者の使命だと断言できる。とにかく「サイバリオン」が，それだけのプレイに耐えるゲームだということは間違いないだろう。

潰えし龍の力

しかしこういった，高度なリピート性を意識して作られたにもかかわらず，実際にゲームセンターにおいてサイバリオンが空前絶後のヒットをしたという記録や記憶はない。むしろ，期待どおりの売り上げを得られなかったとして，営業的には失敗したという，よくある三日月型ゲームと同じ見方が強いのが現実である。こういったゲームがもっている理想と裏腹な結果を生んでしまう原因は，リピートの欲求を起こさせる要素である選択性や難度のシステムにおける，規模の不明瞭さと，不適当な大きさとではないかと考えられる。

いわゆるルート分岐型のゲームでは往々にして，その規模(全体マップ)が示されている。しかし「サイバリオン」のサブストーリーでは，そういったものが作れないほど大きく複雑であるし，またその展開にプレイヤーの意思が反映できないという構造的な問題も抱えている。意図的に制覇していくことが困難で，なおかつ覚えきれないほどの規模では，気力のほうが先に萎えてしまっても，責めることはできないだろう。全体として魅力的にもかかわらず，個々のプレイにおける経験の比重が小さく，多くのプレイが必要である点が，諸刃の剣のような問題を生み出しているのである。

そして難度についても同じようなことがいえる。高度なプレイヤーに到達感や達成感を意識させる仕組みばかり深く，かつ細かくなっており，初心者や中級者にはそういった恩恵が受けられない。確かにゲームプレイヤーにはさまざまな人種がいて，なかには途方もなく上手いといわれる人がいるが，上級者よりは中級者，それよりさらに初級者のほうが多いはずである。商業的にそういった構造を無視してしまうと，印象が悪くなるのは仕方がないというしかないだろう。

こうした仕掛けのゲームは，どう考えてもじっくりと取り組むことのできる家庭用やパソコン向けのような印象があるのだが，アーケードゲームに珍しくないのが不思議である。最近でもノーミスノーボンバーで隠しステージにいけるとか，総合トップのポイントだと2周目スタートなどの上級者を優遇したフィーチャーを耳にしたことがある。しかし，やはりそういった部分の魅力は，商品のメリットとしては評価されていないようだ。

不死不滅の龍

いままで書いてきたことをふまえて，「サイバリオン」をプレイしようとしても，現在となっては入手が困難なソフトになっているようだ。元のアーケード版のファン層の狭さから，X68000版もそれほど大ヒットしたようにも見えなかったし，グラフィックの違いで印象を悪くして躊躇した人も決して少なくないだろう。

それでもグラフィックと操作系以外はサウンドもがんばっているし，動きも悪くないので，全体としてみると悪いゲームではない。むしろ今回考察してきた三日月型ゲームとしての側面は十分に再現されているので，実体験するにはもってこいだといえるだろう。たとえ現在のゲーム市場で商品として時代遅れになっていたとしても，この「サイバリオン」は，多くのことをプレイヤーに教えてくれるはずである。細ければ細いほど美しい三日月の隠れた闇の部分を知る楽しみは，一度覚えるとクセになってしまうかもしれないので，ちゃんと覚悟してからプレイすることを忘れないよう，最後に忠告しておこう。

始まりの年1987

Nakano Shuichi 中野 修一

X68000のゲームには名作が多くあったことは疑いのないところである。個々の作品のデキのよさもさることながら，ユーザーに圧倒的な支持を得たいくつかの作品については，その時代の背景を抜きにしては語れない部分がある。そこで名作ゲームを振り返ってレビューするとともに，このコーナーでは年代順にゲームの発売状況や時代背景などについて考えてみたい。

とりあえず第1回目の担当は私，中野修一でお贈りする。

＊　＊　＊

まず，X68000最初の年，1987──。今月は1987年に発売されたゲームとそれを取り巻く環境を思い起こしてみたい。

遡って1986年を見てみよう。

世間では8ビットから16ビットへの移行がほぼ終わり，PC-9801VMシリーズの躍進が見られた頃である。一方，シャープは8ビット機でふんばり，X1turboZなどを発売していたものの，CPU周りの基本性能は初代X1と変わらない状況であった。周りの高性能新機種ラッシュの中，取り残された状況に倦みX1に続く新世代機が待望されていた時期である。まあ，いまほどではないにせよだ。

メーカーがまったく動く気配を見せぬため「シャープに爆弾を仕掛ける会」発足なんてのもあった。ユーザーの状況としては現在よりもラディカルなものだったかもしれない。そうこうするうちに「シャープが16ビット機を開発しているらしい」という噂とも願望ともつかないものは徐々に広まっていった。

とはいっても，発表直前まで情報は堅く閉ざされていたので，あちこちでさまざまな憶測を生んでいた。

Oh!X編集室も例外ではなかった。当時の安田編集長はときたまシャープに出かけていって渋い顔で戻ってくることがあったようだが（たぶんビジュアルシェルのサンプルを見てきたのだと思う），端からではどんなものが進んでいるのかはまったくわからなかった。

愛読者ハガキなどを見ても，全体的に「CPUは68000しかない」という雰囲気はあったものの，V30説も結構しぶとく残っていたように思う。まあ，所詮は噂であるが。

そうこうするうち，たいていの者の願望を上回るスペックを持ったマシン，X68000がエレクトロニクスショウで劇的なデビューを果たしたことは記憶に新しい。

数々のハードウェアスペックを誇っていたが，それらと同等以上に人々を圧倒したのは，付属ソフトとしてグラディウスが予定されているという事実，当時の「アーケードゲームと同じスペックを持ったパソコン」という空前絶後のジャンルを開いたということだろう。

無論，ゲーム以外の部分の機能も当時の常識を遥かに上回るものではあった。X68000がゲームにこだわっていたのは，むしろ，全体的なパフォーマンスのよさをアピールするためのことであったようにも思われる。さらに「ゲームもできるパソコン」という捉え方は，ほとんどの場合イメージ的にマイナス要因にしかならなかったようにも思われる。

しかし，「アーケードスペックのパソコン」というフィーチャーが，夢を託すに足るものとして人々の心を大きくつかんだ特徴であったのは間違いない事実であろう。

1987

発表は1986年の冬，そして発売は1987年の春。X68000の歴史はそこから始まっていった。発表から半年という長いおあずけの期間を経てようやくの発売であった。ようやく今回の本題に入ることにしよう。

＊　＊　＊

少則得

多則惑

先哲のいう，「少なければすなわち得る，多ければすなわち惑う」という言葉は，ものが少なければ，新しいものを得たときの喜びも増し，ものが多ければ迷ってどれも中途半端にしか楽しめない，といった状態を表すものだ。

X68000における最初の1年は，まさにこの言葉の示すところを如実に表していたといっていいだろう。

1987年に発売されたゲームソフトの本数自体は非常に少なかった。わずかに6本である。だからといって悲観してしまう人はいなかったし，別にPC-9801で出ているようなゲームが揃ってないからといって前途を心配する人もいなかった。ハードウェア的なスペックは従来のパソコンとは異次元のものであったし，第一，みんな前しか見ていない状態だったのだ。

グラディウス

最初はグラディウスしかなかった。

説明の必要もないだろうが，グラディウスはパワーアップ型シューティングゲームのかたちを確立した名作アーケードゲームである。プレイヤーの装備によって敵の攻撃が変化するオートレベルコントロール，強力な武装と小技の利くオプション，多彩な面構成と魅力は尽きない。

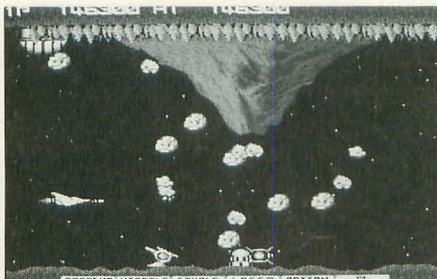
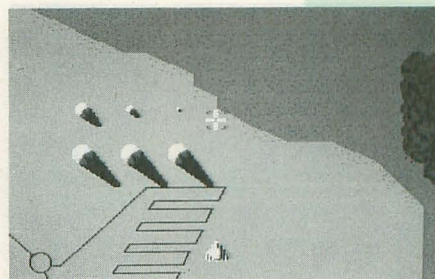
X68000発売以前，OSなどが完成する前からグラディウスは動いていたので，編集室に届いたX68000試作機で日課として1周をこなす人も多かった。こう書くとなんでもないことのようだが，裏話として，最初はOS用BIOS ROMをグラディウス用BIOS ROMに抜き差ししてやらなければならないため，グラディウスをやるたびにフタを開けてROMを交換し，ほかの作業をするときはまた開けて組み直すという結構面倒な作業が必要だったのだ。

最初はみんな初心者だったので動きの滑らかさなどにスプライトの力を感じるものの，まだどう凄いかはよくわかっていなかった。やがて慣れてきたとはいえ，スピード2でやるのがせいぜいで，1周クリアするとしても慣れない人はスピード1でやっている状態だったのだ。

やがて「ゲーマーな人」のプレイを目のあたりにすることになる。スピードを上げ，大きく散開したオプションから華麗に舞うレーザー……オートレベルアップで敵の攻撃もこれまでになく激しく過熱していく。

ゲームセンターでのグラディウスをまったく見たことがないわけではなかったが，くすんだゲーセンのモニタで見るのとは別世界の美しい画像がそこにあった。ほとんどX1やMSXでのグラディウスしか知らなかった身には滑らかで美しい映像に思わず，

[1987]

スピード3ではオプションも広がる

第2関門の逆火山

横長画面が悲しい

「これがグラディウスだったのか！」
ともらすと，
「そうです。これがグラディウスなんですよ」
と答えが返ってくる。華麗で緻密なアーケードゲームというものの実力をまざまざと見せつけられた思いであった。

素人が普通にプレイをしていても2周目3周目の攻略などは思いもつかない。が，そのとき目の前に展開されていたのは，まさにそういう世界だったのだ。最初思っていたのよりさらに懐が深いゲームだということも思い知らされた。

アーケードゲームはパソコンやファミコンとはゲームの質が違う。現在ではゲーム機の性能も上がっているのでこれくらいの処理は当たり前にこなしてしまうことはわかっているが，じゃあこれと同じくらい面白いゲームがゴロゴロしているかというと，そうでもない。

アーケードゲームの数あるなかで，同梱ソフトにグラディウスが選ばれたというのは素晴らしい幸運だったのかもしれない。

当時，編集の@氏に，
「これくらいのが年に2，3本出てくれればいいんですが」
といったら，
「このクラスなら年に1本でもいいよ」
と切り返された。実際，発売前から半年はそれ1本やり込むだけで皆十分堪能していたのだ。

＊　＊　＊

とはいうものの，実は，私自身は長い間グラディウスには手を出さなかった。「やっぱり必修科目かな」とグラディウスを本格的にやり始めたのはX68000発売から2年近くたってからになる。その時点で再確認したことは，すでに発売されていたほとんどのゲームより面白いという事実であった。最強装備をキープしての攻防と各面での復活がそれぞれ熱い。まさにアーケードの名作ゲームというにふさわしい。

久しぶりにやってみると結構きつい。発売されて最初の頃は，
「モアイが越えられません」
という悲痛な声も多く聞いたものだが，やがてそれも収まっていった。皆，精進していたようだ。もともと下手な人は下手な人なりに，うまい人はうまい人なりに楽しめるゲームではあったのだが。

しかし，あれをちゃんとこなすようなユーザーが揃っているとすると，ゲームメーカーにとってはちょっとした脅威だったかもしれない。

ゼビウス

いわずと知れたナムコの名作シューティングである。

非常に早い時期に発表された移植作だが，評判はあまりよくない。MZ-2500版のほうがマシという説もある。

縦画面シューティングを画面いっぱいに変形して表示しているのだが，敵弾の速度などで横方向の移動は遅いが，縦方向は異様に速いといった問題があった。

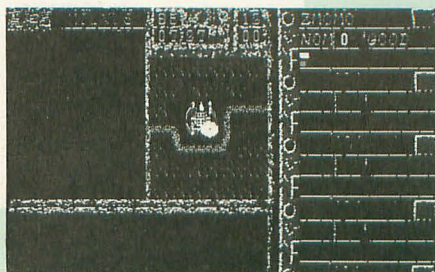
ある種の縦画面シューティングゲームをX68000に移植するにはどうしても，こういった問題を回避することはできなかった。その後に発売された縦シューを見てもそれぞれに苦労していることがわかる。単に上下を切っても敵の出現パターンが同じなためゲームにならなかったり，自キャラの動ける範囲が狭くなってしまっていたり，ドラゴンスピリットのようにモニタ横倒しモードをつけるという手もあるが，なかなか難しいところである。

かなり表示エリアは小さくなってしまうものの理論上最適となる横256×縦384，正方形ドットといった画面モードが発見されたのはずいぶん後のことになる。この時点でひとつの本質的問題がすでに露顕しているわけだ。

そのほかの問題点もあった。一応ひととおり動いていて，確かにゼビウスなんだけど，キャラクター出現の法則性だとか性格だとかも結構いい加減だったように思う。いまひとつ気合に欠ける

魔神宮

X68000用RPGの第1弾。ご存じザインソフトの作品だ。基本的には正統派のファンタジーRPGで，結構細かくキャラクターメイキングを行ったりしているのだが，延々と続く乱数での敵発生，コマンドでの戦闘と正統派すぎてうざったい面もあったように思う。

X-BASICで書かれていたので起動にはしばらく時間がかかるが，ゲームの進行自体はそれほどには遅くない（もちろん速くもないが）。

これが起動直後の画面

戦闘画面。つらい……

主人公キャラがマップ上ではなにか「もやもやした雲」のようなもので表されており，いきなりなにが始まったのかわからなくなるという代物だった。最初に見たときはサンプル版だからキャラクターができていないのだと思っていたら，結局そのままだったので非常に驚いた覚えがある。いまもって，なぜこのようにしたのか意図がわからない。かなり斬新な表現ではある。

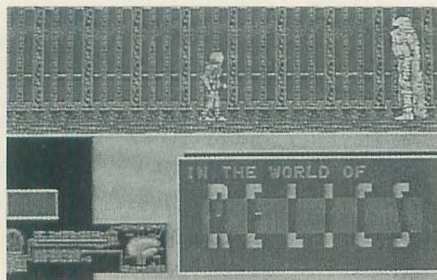
レビュー中にバグが出て止まってしまったのでプログラムを修正してメーカーに教えてあげたら逆に怒られたとかいう逸話もあった。そんなこともあってか，このゲーム，まだ解き終わったという人の話を聞いたことはない。

これ以降，X68000オリジナルのRPGがいくつか発売されているが，X68000ではRPG自体いまひとつヒット作に恵まれていない。ある程度売れたものというと桃太郎伝説，ダンジョンマスター，イースくらいのものだろうか。オリジナル作品となると印象に残るものは少ない。

レリクス

完全移植というやつですね

多関節キャラを使ったリアルな動きが売りのアクションアドベンチャーゲーム。もともとのゲーム自体は独特の雰囲気を備えた秀作で，PC-9801や8ビット機で複雑な動きのキャラクターを導入した志の高いゲームだったのだが，X68000では640×400ドットの小さな画面，8色のグラフィック，のたのたとした動き。グラフィックへの書き込みを行ったにしても，もう少しやりようはあるんじゃないかと思える。

いまにして思えば「プリペル」とかよりはまともかもしれないという気もするが，「グラディウスが基準」という当時の風潮のなかでは受け入れられるべくもなかった。特に祝一平氏の逆鱗に触れ，「尼寺へ行け！」とお叱りを受けた作品であった。

ここで出てきた問題は，これ以降，他機種からの移植を行う際全般に必ず表れてきた問題でもあった。X68000にはグラフィック，テキスト，スプライトと3種類の表示資源があったのだが，どれをどのように使用するかということが重要なのだ。

どういう選択を行った場合に十分なパフォーマンスが得られるか，逆にハードウェアの特徴を出すにはどれだけの変更が必要か，どの程度のものをユーザーは期待しているのかということを見極めなければならない。

レリクスでは画面サイズを640×400のままで移植したことが敗因のひとつだったといえるだろう。まあ，これならばグラフィックの描き直しも必要なく，手軽に作れるのは確かだが，それだけ手をかけなかったことがそのままクオリティに反映されてくるのも確かなのだ。

512×512にしておけばスプライトが使えたのでゲームはまったく別物になったであろう。そうなるとゲーム内容自体がもの足りなくなるおそれも多分にある。難しいところではあろう。

ちょうど退職間際だったプログラマが作り捨てていったとかいう噂も聞いたが，こういうまともな仕事のできない人を抱えたメーカー側も災難ではあった。この反省もあってか，発売元のボーステック自体はその後の銀英伝シリーズでは，他社のベタ移植を後目にかなりよい仕事をするようになったのは不幸中の幸いというところだろうか。

その後，時代を経て「98プログラマにスプライトマネージャが作れるわけねーだろ」とか「XCを使っといて性能を語るな」とか「最適化オプションがきかないなら分割コンパイルくらいしろ」とか「割り込み禁止でDMA転送するくらいならソフトウェア転送にして」といったX68000の常識部分で文句をいいたくなるような作品もいろいろと出てくるわけだが，どんなハードウェアもプログラマ次第ということを痛感させてくれた一作だったといえるだろう。

スペースハリアー

アーケードより移植された疑似3Dのシューティングゲーム，唸りをあげて飛んでくる巨大キャラを撃ち落としつつひたすら前進を続ける。グラディウスに続きX68000の底力を見せてくれた歴史的なゲームである。ざっと考えてユーザーの4人にひとりくらいが買っていたと思われる化け物ゲームだった。

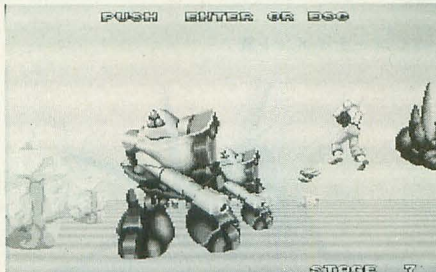
思えばこのゲーム，ゲームセンターで初めて見たときには「ついにここまできたか」という思いがあったものの，操縦桿式のアナログスティックとトリガーボタンに馴染めず，ゲームとしての印象はあまりよくなかった。私にはあんなものを連射することは不可能だったのだ（普通のボタンもあったけど）。

X68000版はデジタルスティックで操作するように変更されている。さらに，たとえ邪道といわれても連射装置なしには語れないゲームだと思う。

移植はまずまずのデキで，地面のチェック模様は省略されたものの，凄まじいスピード感や迫力は十分に再現されていた。完全移植というのとは少し違うが，連射受け付けが非常によく，秒間30発以上の連射にも対応していた。多少タイミングを調整することが必要ではあったが，高速表示される敵をドカドカ撃ちまくる，オリジナルとは別物の爽快なゲームに変貌していたといっていいだろう。

サンプル版を見たシャープのX68000開発者が驚いたという話も伝わってきている。あれだけ大きなキャラクター（それもひと

[1987]

多関節の巨大なボスキャラ

ジェットストリームアタック！

これがX68000版オリジナルのボス

つや2つではない）をあんな速度でグラフィック画面に書き込むことができるとは思っていなかったようだ。

私が特に評価しているのはコンティニューの扱いだ。X68000版スペハリでは自機数が続く限りはオリジナルどおり「その場コンティニュー」，1コイン終了時には一定の面数に戻されるという方式になっている。特にアーケードゲームの移植は完全移植をもって尊しとされる風潮にあり，ゲームシステムの変更を嫌う人もいるが，スペハリの場合，特にパワーアップなどもないのでこのようなシステムにしてもあまり問題はなかったようだ。

ゲームセンターで100円分遊ぶのならともかく，それなりの金額を払って買ったゲームでどうしても先に進めなくなるという状況は好ましくない。誰もがエンディングを見れるのが理想である。だからといって，なにもせずにクリアできてしまうのではやっていて面白くない。そのあたりのバランスがアーケードゲームの移植作の課題でもあるわけだが，スペハリの場合，初心者救済措置としては連射装置の装備だけで十分だったといえるだろう。

その後に発売された作品ではその場コンティニューでもないのにコンティニュー制限がつけられていてユーザーの激怒を買った作品や，なんとなくやっているだけでクリアしてしまい拍子抜けしてしまう作品もあったのだが，たとえば私はその場コンティニューをどんどん重ねていけるアフターバーナーにはあまり入れ込めなかった。達成感が感じられないからである。私がスペハリで初めてハヤオーを倒したときには鼓動が聞こえそうなくらい脈搏増加，手から流れ出た汗でスティックはべとべとになり，もうアドレナリン分泌しまくり状態だった。個人的に，スペハリをこういうバランスにまとめてくれたことには非常に感謝している。

発売後もスペハリは一定の人気を持続した作品であった。それを示す最大のものはなんといってもキャラクターを描き換えた海賊版が存在したことだろう。もともと荒唐無稽な世界ではあったのだが，パロディ化されたキャラクターたちは凄まじい迫力をもってプレイヤーに迫ってくる。

こうして最初のひとりが描き換えると，やがてあちこちで奇怪なグラフィックが飛び交うことになっていった。独特のノリでやたら気合が入っているものが多く，ひそかにアングラ文化が花開きつつあった。

その影響からか，その後発売されたアフターバーナーには最初からキャラクターエディタが付属していたのだが，自分でパターンを描き換えたという話は聞いたことがない。

大ヒットという点ではアフターバーナーに及ばないと思われるが，ユーザーの思い入れの大きさではスペースハリアーはまさに「歴史的」ゲームといえるものだったのだ。

画面はそれほど派手ではない

マンハッタンレクイエム

X68000初のアドベンチャーゲーム（まあ，この時期はなにをやっても初になるのだが）として登場したこのゲーム，いま思えばグラフィックなども地味めなのだが，使っていて「おっ！」と思わせるハイタッチな仕上がりのアドベンチャーゲームだった。

ゲーム内容自体は他機種版と差異はない。プレイヤーは探偵J.B.ハロルドとなって殺人事件を捜査していくのだが，やっていると「ここはこう見せるべきだ」とか，かっこいいユーザーインタフェイスのあり方とかがわかっている開発者によって作られていたことがよくわかる。X68000だとアドベンチャーゲームはこうなるという明確な主張がなによりもうれしかったことを覚えている。

当時，3千本ちょいの売り上げだったはずだが，メーカーが「こんなにたくさん売れたのは初めてだ」と喜んでいたという話を聞いた。いかにもアドベンチャーゲーム不遇の時代を思わせる。

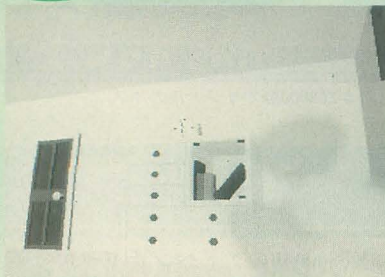
リバーヒルソフトの真骨頂は，次第にパワーアップしながらこのあとに続く一連の作品なのだが，それはまた別のお話。それらについてはまた次の機会に紹介することにしよう。

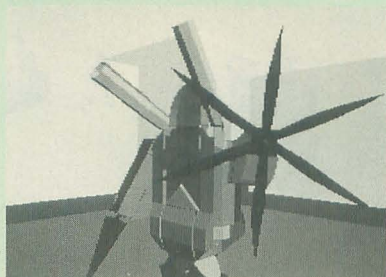
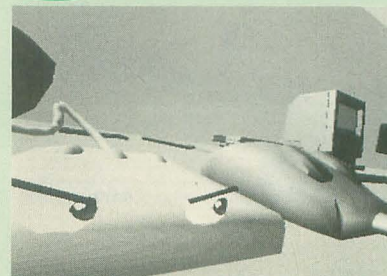
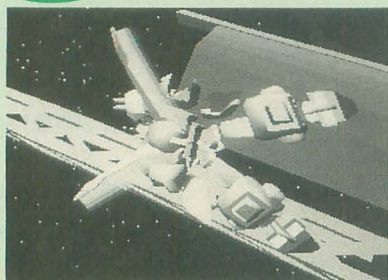
操作感はかなりよかった

Graphic Gallery

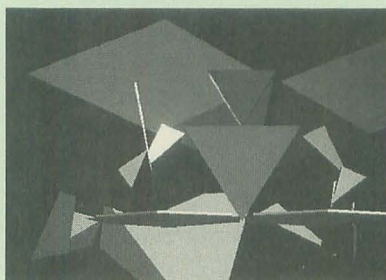
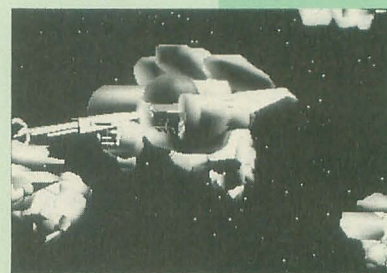
連載ではCGA作品の内容のつけ方，特にストーリーのふくらませ方について考えています。「Graphic Gallery」では，連載で取り上げられた過去のアマチュアCGAコンテストの作品と，そのほかの入賞作品を紹介します。

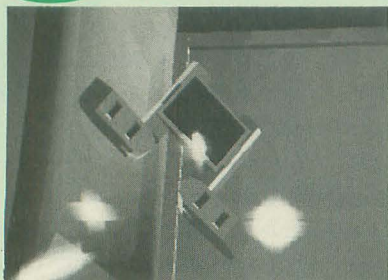
第3回

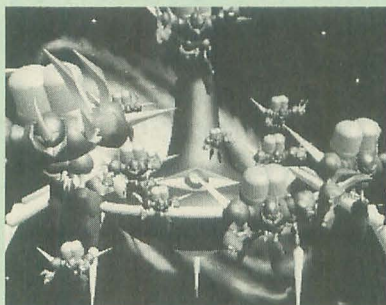
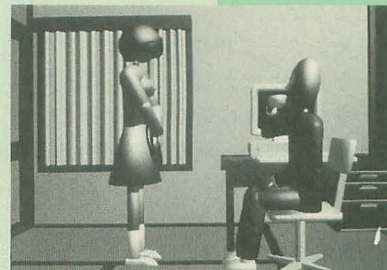
最優秀技術賞・作品賞

CLOCK 上原 哲太郎(KMC)

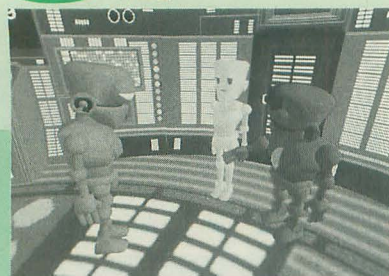
エンターテイメント賞

ゲッピーロボ 石井 源久(KMC)

第4回

エンターテイメント賞

DesperadO 篠田 直樹(KMC)

映像賞

Love is the message
田中 政斗(ぺしぺし企画)

第6回

作品賞

Switch On Concent Robo
坊野 博典(KMC)

エンターテイメント賞

冥王龍ペルギウス 腰原 仁志

第7回

作品賞

LowReso 中村 ゆういち

作品賞

電神 ギガダイン 腰原 仁志

第5回

努力賞

MOUSE 京大マイコンクラブ

アクション賞

MISSION 浅野 英史

佳作

ある夜の出来事 小島 禎樹

入選

歩行者とドライバーとのコミュニケーション
太田 敦司

X68000にQuicktimeのような統括的な映像環境はついに登場しなかったが，ここに至ってようやく新しいアプローチが可能になってきた。映像再生環境としてはともかく，制作環境としては満足できるものが実現できることがわかったからだ。
SCSI2ボード1枚でなにが違うのか？　SCSI2ボードとはいっても，機能自体は内蔵のSCSIとほとんど変わるものではない。速度が速くなっただけの話だ。ここではまさに速度自体が問題だったのだ。256×256ドット，65536色の画像を秒間20コマ再生するためには，2.5Mバイト/秒の転送速度が必要だったわけだが，それがクリアされるようになってきたのだ。幸か不幸かX68000のG-RAMが24ビット構成でなかったことも幸いしている。
アニメーションの無圧縮保存というのはそうとうに無茶な話だ。よほどのデータ転送速度がなければやっていけない。転送速度が稼げないから世の中では圧縮が行われる。人一倍転送速度が遅い状態なのに無圧縮データ以外考えないというのは怠慢である。
これまでは映像制作環境となるとメインメモリ上からの圧縮展開再生以外に道はなかった。完成画像の一括再生が不可能で，最終的につなぎ撮りという手作業を経るため，統合的な制作環境は作りにくかったといえるだろう。これでは編集作業に手間がかけられるはずもない。
今回のHDアニメーション法には環境を厳しく特定し，犠牲にするものも多くあるのだが，ハードウェアの不備を補うためにはこれもいたしかたない。これまではどうしてもX68000ベースで考えることが必要であったため制限が多かったのだが，今回は上限ぎりぎりのところを狙っている。そしてなんとか実用になるフォーマットを確認できたと思う。特殊なハードウェアを使用しない限りこれ以上のものは不可能である。X68030はCPUとしてはすでに時代遅れの感があるものの，秘めたポテンシャルは高く，むしろ多くの機能が閉じ込められている状態である。今回確立されたシステムの実用度はかなり高い。あとの問題は音声同期だけなのだが……。

CONTENTS

序論　基本環境の確認

アニメーションの現状

Nakano Shuichi 中野 修一

**もともと映像に強いマシンとして生まれてきたX68000
映像分野でさまざまな成果をあげているのは事実である
その陰でX68000の抱えてきたさまざまな問題点を見てみよう**

プロジェクトチームDōGAの活躍があってか，X68000ではアニメーション制作という用途がかなりメジャーなものとして存在している。しかし，その基礎環境はというと，お世辞にも整備されているとはいいがたいものがある。

細かい部分で見ればいろいろ整備が進んでいるのは確かだが，全体的に見ればかなり面倒なことを繰り返してなんとか目標に到達している節がある。そのような環境で作品を作り出す努力には敬意を払いたい。

ここではアニメーション制作の基礎環境から見直しを行ってみよう。どのような環境が望ましいのか，現在のX68000で可能なこと，どのようにすればどこまでできるのかについて検討してみた。

X68000におけるアニメーション環境

まず開発されたのがANIM.Xであった。これは65536色画像をランレングス圧縮でメモリに蓄え，リアルタイムに展開表示するシステムであった。再生レートは秒間15コマ（15fps）が基本となっている。

やがて，画像を256色に減色し，2ドットを同時に書き込むことで高速化されたHANIM.Xが登場し，ほぼ標準の地位を占めることになる。これで20fpsのプラットフォームが確立された。

いずれの方式もメモリ上に圧縮データを置いて一気に展開することから，1回あたりの再生時間はメモリ量に依存し，メインメモリ12MバイトしかつめないX68000シリーズではどうしても煩雑なつなぎ撮りを必要とした。

このシステムでの問題は256色への減色処理とメモリ量であったといえるだろう。256色ではどうしてもマッハバンドが目立つ画像になり，それを回避すべくディザリングを導入するとデータが肥大化してメモリを圧迫し，再生速度も遅くなるという結果をもたらした。X68030の登場である程度複雑な画像でもスムースな再生が可能になったのは確かだが，本質的な解決にはなっていない。

こういった問題の解決策として，2台のX68000をつないでフルカラー画像を出力するというシステムが提案され，画質の確保は達成されたものの，細かなつなぎ撮りを回避することはできていない。

非力なCPU上を使ってソフトウェアでアニメーション再生を行う場合，ランレングス圧縮というのは有効な方法であった。特に初期のDōGA作品の場合，フラットシェーディングポリゴンにテクスチャマッピングなしという映像が主体であったので，ランレングス法による圧縮は順当な選択だったといえるだろう。

その後，レンダリング技術の発展により，CGAシステムの表現力はどんどん上がっていった。まずテクスチャマッピングが導入され，続いてスムースシェーディングや点光源が実現されていった。さらにディザリングが可能になり，こうなるとランレングスはまったく有効な方式とはいえなくなってしまっているのだが，これに代わる方式はまだ導入されていない。

他機種の状況

DōGA CGAシステムの場合，最終出力がビデオテープになるので，作業量よりもまず画質が優先される。

Macintoshなどの場合，Quicktimeを使用することになる。AT互換機などWINDOWS勢もそれがVideo for WINDOWSあるいはQuicktime for WINDOWSになるかの違いはあるもののほぼ似たようなものになる。これらのシステムでは圧縮方式の選択は任意だが，最終的にCD-ROMが媒体になることが多いので，そうなると圧縮方式はほぼシネパックに限られることになる。

詳細は別記事に譲るが，これは画像の変化点のみを書き換える方式だ。YC分離を行い，輝度情報は2×2単位，色情報は4×4単位でチップ化して置き換えを行う。デコードは軽く，エンコードには非常に時間がかかることでも知られている。書き換え場所が少ないのでVRAMアクセスを抑えることができるという利点もある。

そのほかの圧縮方法も用意されているが，基本的にディスクからリアルタイムに読み出し，場合によっては圧縮ファイルを展開しつつ画面に表示するというシステムである。圧縮/展開プログラムは差し替え可能でシステムは主に画面管理，時間管理などを担当する。

Macintoshなんてのは最近映像よりになってきただけで，ずっと以前からアニメーション分野で勢力を伸ばしていたものにAMIGAがあった。

これもエレクトリックアーツによるIFFフォーマットの確立が非常に大きな意味を持っていた。現在に至ってもDeluxePaintに比肩する先見性を持ったグラフィックツールの存在というのは見受けられない。

AMIGAの動画はオンメモリのデータを展開しながら再生というDōGA形式と似た感じのものになっている。使用できるメモリ量，VRAM形式，使用色数やデータの圧縮効率が違うので一概にはいえないが，かなりの長さのものを一括再生することができるシステムである。

ANIMファイルの構成は，大まかにいうと2画面前の画像との排他的論理和を圧縮するという方式だ。

模式的にいえば，

前の画面	差分	次の画面
0	1	1
0	0	0
1	1	0
1	0	1

のようになる。前の画面と同じ絵があれば

そこには0が続くことになるのがわかると思う。これをAMIGAでは1バイトずつ縦方向にランレングスを取る。AMIGAのVRAMは水平型なので8ドット幅で縦にスキャンするかたちになる。ランレングスならむしろ横方向のほうがプログラム効率がよくなることはわかると思う。なぜ縦方向にしているかというと，縦方向のほうがランが続きやすく，圧縮効率が上がるからである。画面に任意の正方形を描いて試してみればすぐにわかると思う。

で，こういった差分形式のもの(たとえばMPEGやCINEPAK)ではデータ量を減らすことができるものの，途中からの再生とか，逆再生といったトリックプレイに対応するのが困難だという問題点も抱えている。

IFF形式アニメーションの巧妙さは，排他的論理和を使うことでデータを逆方向の差分としても扱うことができるという点であった。いわゆるPingPongPlay(往復再生)はAMIGAでは基本的なフィーチャーであったのだ。

圧縮再生は可能か

では，X68000でシネパックあるいは圧縮ファイルをディスクから読み出して展開するというシステムを採用することは可能だろうか？　HANIM.Xはメモリからの展開表示だけで限界近い状態なのでHANIM.Xと同じパフォーマンスを期待することはできないことはわかる。

CPUパワーは040turboなどで多少は補うことができる。ディスクからの読み込み速度もSCSI2ボードを使えば4倍に上げることができる。やろうと思えば結構いい線までいきそうな気がするのだが，今回はこのアプローチは採用していない。現段階で要求されている複雑な画像ではランレングス圧縮は無意味だからだ。

ランレングス圧縮はVideo for WINDOWSでも採用されている方式のひとつだが，詳しい人に聞いても「他方式を目立たせるために入れてるんじゃない？」といわれるくらい現在では評価が低い。とりわけ，テクスチャに弱いというのが致命的である。いまやパソコンのリアルタイムゲームあたりでもテクスチャマッピングは当たり前のように行われているのだから。さらに取り込み画像などを相手にすると，使わないほうがマシなくらいだ。

しかし，高画質で，しかもソフトウェア再生で高いフレームレートを確保できる方式，しかもできれば10MHzのX68000で実用的なもの，という条件に当てはまる方式はまず存在しない。

アニメーションデータというのは膨大な情報量を持っているため，それを扱うには非常に多大な処理能力を要求される。

MacintoshにしてもQuicktimeを導入する際に16ビット機を切り捨てている経緯があるのだが，16ビット機が主体のX68000ではこれは難しいところである。

さらに，シネパックなどでは時間管理はするものの，前の画面との差分を取っている関係上，無制限にコマ落ち表示させるわけにはいかないという問題がある。基本的には速い機種なら滑らかに，遅い機種ではカクカクと表示することで互換性を保つシステムではあるが，それにも限界があるということだ。表示時間のウェイトをはずし，メインメモリからVRAMへの転送を省略するのがせいぜいだ。ある程度以上のCPUパワーあるいは付加的なハードウェアがないと話にもならない。

フォーマットの確立

ということで，「DōGAクラス」の映像を得るための長時間デコーダは難しいということなのだが，現状にはそれ以外にも問題点がある。DōGAでは加工の手間などもあってか，アニメーション映像というのは制作環境にべったりと密着しており，ひとつのアニメーションをひとつのファイルにまとめるということを行っていない。

同じオンメモリシステムでもAMIGAがアニメーションファイルを流通できる形態で提供していたのに対し，X68000ではPANIC以外にアニメーションの流通がほとんどなかったという事実はこれに起因しているように思われる。

ビデオ作品制作という目的もよいが，それ以外にもアニメーションというのは使用価値のあるものである。現在ではたくさんのアニメーションファイルがあちこちに流通している。それらのビュアがないというのは非常にまずい。インターネットでもアニメーションファイルはQuicktimeで統一されていく傾向にある。最低限Quicktimeファイルのエンコード/デコードはメーカーにサポートしてもらわないとどうしようもない。

これらの問題点をひと言でまとめると「ファイルフォーマットがない」ということになる。

昔からそうだったのだが，シャープはフォーマットを作ることが下手である。X68000の登場時も512×512ドット65536色表示というハードウェアスペックを持っていたのだが，そのデータを保存する方法については，ベタ形式しか考えられていなかった(なにも考えられてなかったともいう)。16色，256色データのパレットに関しては「保存する」という概念自体がなかったように思われる。

ハードウェアがどう使われるか，データをどう使うかという部分で考えていけばフォーマットは自ずと確立していく。まず，そこから考えていくという意味ではアップル社は凄い。そしてシステムに無理なく取り込む手際も凄い。こういった大規模なインフラ整備はハードウェアメーカーでなければなかなかできるものではない。

SX-WINDOWのCGAファイルが一応その役目を果たすものではある。どうしてこうわけのわからないことをするのか不明だが，なんと「ファイルフォーマットしか」決めていない。最大の問題はSX-WINDOWでグラフィックを扱うこと自体がナンセンスだったという事実だろう。基本にあるのが，メモリの苦しいSX環境で，常識で考えられる量の数倍のメモリを必要とし，常識で考えられる10倍の時間をかけて表示を行うシステムでは使えというほうが無理というものだ。

CGAファイルの構造は正式には公開されていないが，考え方自体は柔軟で決して間違っているわけではない。1枚絵を並べて時間管理を行いながら再生する(音は垂れ流し)というものだった。1枚絵部分の圧縮にはAPICからJPEG，TIFFなどまでサポートされていた(要するにIVMのサポートする画像フォーマット)。JPEGを選択しておけば「MotionJPEGをサポートしていた」といってもさほど間違いではあるまい。システムとしては結構美しい。システムの演算ドライバに無限桁演算ルーチンを採用するのと同じくらい美しい思想だ。

これが実用的な速度で動けば問題ないのだが，当のIVM自体が1枚絵を表示するのにも信じがたいくらいの時間と莫大なメモリを使う。実はIVMはLISPを内蔵していて画像デコーダはインタプリタで動いているという噂が立つくらいの凄まじさだったのだ。まあ，CPUパワーが3桁分上がって，メモリが無尽蔵にあれば問題はなかったのだが，当時すでにX68000シリーズはどちらも業界最低クラスを独走していたのは周知のとおりであった。

このような状況の中，現在実現できるアニメーション環境をもう一度考えてみよう。

[特集]

各論1：アマチュアCGA学会番外編

SCSI2を使用したHDアニメーション

Takatsu Masamichi 高津 正道

期待のSCSI2ボードを最大限に活用する
HDアニメーションへのアプローチとしてDōGAからの助っ人の登場だ
無圧縮再生表示のための基本ツールを提供する

はじめに

あの満開製作所からX68000用の高速SCSI-2ボードMach-2（定価28,800円）が7月に発売されました。SCSI-2ボードとはいかなるものであるかとか，製品の詳しい内容につきましては，先月号をご覧ください。

早速，我々も試作機を入手いたしましたので，最高5Mバイト/秒という転送速度を生かしたハードディスクアニメーションの可能性を探ってみることにしました。

Set up

まず，PL法の影響でしょうか，使用上の注意として，

・データが飛ぶかも。バックアップを忘れずに。

・相性の悪いHDDがあるかもしれぬ。よく吟味すべし。

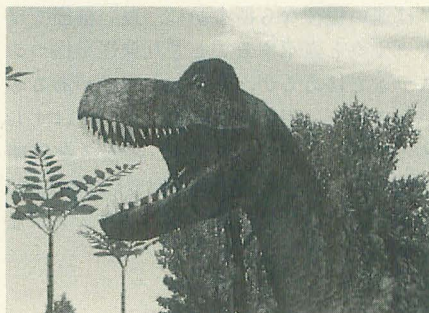
なんて書いてあります。そこで，とりあえずHDのバックアップを取ってから，このMach-2をX68030（33MHz改）に装着しました。

接続したHDはQuantum LT540Sです。そんなに速くないドライブですが，それでも現行機種ですので，それなりの速度が出ると期待されます。読み込み速度をdskbenchを用いて速度を測ったところ，

	内蔵	Mach-2
Sequential	910.2[KB/s]	3276.8[KB/s]
Random	728.1[KB/s]	1456.3[KB/s]

となりました。

内蔵のHD，つまりただのSCSIの場合と比べて，2～3.6倍程度速くなっています。この場合の最高速度である3276.8Kバイト/秒という値も，Mach-2の性能を出し切ったものではなく，どうやらLT540Sの転送能力が最高3.2Mバイト/秒だということのようです。

これだけの速度が出るのであれば，256×256ドット，65536色で，秒あたり3276.8/128＝25.6枚のアニメーションが可能のはずです。

AMIデータでも一応再成可能

AMIで手軽にHDアニメ

HDアニメーションというと，本誌1994年3月号に載った，AMIシステムがあります。これが使えればお手軽なので，早速試してみました。

	内蔵	Mach-2
256×256 64K色	6枚/s	6枚/s
256×256 256色	12枚/s	12枚/s
128×128 64K色	20枚/s	20枚/s

しかし，その結果は，Mach-2を使用しても同じ速さしか出ません。これはどういうことでしょう。

おそらくMach-2はG-RAMへの読み込みが遅いのではないかと思われます。ともかく，AMIを使ってお手軽にMach-2の威力を体験するという野望は潰えてしまいました。

G-RAMへの読み込み

次に，HDからメインメモリへの読み込みと，G-RAMへの読み込みでどれだけ速度が変わってくるのか，簡単なプログラムを作って，ベンチマークを取ってみました（List1：bench.c）。中身は，512Kバイトの読み込みを25回くり返して，その時間を計っているだけです。また，このプログラムでは，HDからメインメモリへの読み込み，HDからG-RAMへの読み込みの他に，HDから一度メインメモリに読み込んでからGRAMへ転送した場合についても測定しています。

測定結果は，

	内蔵	Mach-2
HD→メインRAM	906KB/s	2872KB/s
HD→GRAM	906KB/s	797KB/s
HD→RAM→GRAM	758KB/s	1833KB/s

となりました。推測どおり，G-RAMへの読み込みがとっても遅くなっています。

この点につきましての満開製作所の回答は，

「機種によっては，バスマスタでG-RAMに転送すると画面にノイズが乗る場合があるので，わざと遅くしている。製品版では遅くするかどうか選べるようにする予定」ということでした。

そういえば，X68000XVIの一部のロットには，HDからG-RAMに読み込むとデータが化けるものがありました。X68030にもそういうトラブルがあるということなんでしょう。

新しいHDアニメーション

Mach-2を使った場合，G-RAMに直接読み込むよりも，一度メインメモリに読み込んだほうが速くなることがわかりました。その場合でも，1833/128=14.23ということで，解像度256×256，6万5千色で秒15枚くらいのアニメーションならできそうです。

ということで，一度メインメモリに読み込んでからG-RAMに転送する方式のHDアニメーションプログラムを作成しました（List2：hdanim.c）。ついでに，128×128ドットのときは256×256ドットに拡大する機能もつけてあります。gcc＋XC ver.2のライブラリを使用して作成しましたので，li

bcを使用している場合には関数名を変える必要があるでしょう。

使用方法は,

HDANIM[オプション]<ファイル名>

で,オプションは,

-L　　15KHz モードで再生します

-T<n>　1枚あたりの表示フィールド数を<n>にします

-S　　128×128として再生します

-Z　　128×128を256×256に拡大表示します

が使えます。また,アニメーション中は,ROLLUPで再生速度を速く,ROLLDOWNで遅くできます。そしてESCで終了します。

入力ファイルの形式は,単にG-RAMの内容をファイルに出力しただけのものです。圧縮はしていません。

表示については,512×512の1画面を,図1のような順番で行います。この順番はAMIファイルと同じですので,65536色のAMIファイルを指定すれば,1ライン分ゴミが表示されるものの問題なくアニメーションが行われます。

また,アニメーションさせるプログラムだけ載せても使えませんので,データを作成するためのプログラムも載せておきます(List3:gdump.c)。

単に,表示されているG-RAMの内容をそのままファイルに追加しているだけです。オプションもなにもありません。アニメーションさせたい画像を図1の順に表示してからgdumpを実行する,ということを繰り返せばデータファイルが作成できます。

このプログラムを使ってアニメーション速度を測ったところ,

	内蔵	Mach-2
256×256 64K色	5.7枚/s	15.9枚/s
128×128 64K色	25.3枚/s	60.0枚/s
128×128 64K色(拡大)	17.6枚/s	30.7枚/s

となりました。

結論と今後の野望

Mach-2をスロットに装着するだけで,HDの転送速度が3倍近く向上するというのは大きな魅力です。特にHDアニメーションの場合,毎秒6枚と毎秒15枚では格段の違いといえます。

さらに,まだ高速化できる余地があります。今回は時間がありませんでしたが,64K色を256色に制限すれば,解像度256×256で毎秒30フレームも可能なはずです。また,現在問題となっているのは,HDの転送速度ではなく,G-RAMのアクセス速度ですので,G-RAMに書き込む量を減らせばさらに高速化します。具体的にいうと,フレーム間の差分をとるだけです。

もっとも,この方法では,画像の内容によって,最高速度が変わってきますが,通常の場合,256×256の64K色でも毎秒20枚は可能であると予想されます。時間があれば,256色モードや画面差をとることができるhdanim.cを制作し,電脳倶楽部にでも掲載したいと思います。

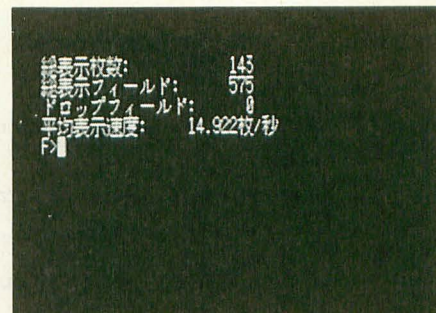
このように,Mach-2はHDを高速化し,X68000の環境を改善するだけでなく,X68000でHDアニメーションを実現しうる性能を持ったボードであることがわかりました。

再生時のコマ落ちが確認できる

図1　表示順序

リスト1　bench.c

```
 1: /*
 2:    HD ベンチマークプログラム bench.c
 3:
 4:    gcc -O -fomit-frame-pointer -fstrength-reduce bench.c -ldos -liocs -
lfloatfnc
 5: */
 6: #include <stdio.h>
 7: #include <io.h>
 8: #include <stdlib.h>
 9: #include <string.h>
10: #include <time.h>
11: #include <doslib.h>
12:
13: #define SIZE (1024*512)
14:
15: #define COUNT 25
16:
17: unsigned char *dst, *src;
18: int fd;
19:
20: #define LOOP 64
21:
22: #define COPY1()         *dst++ = *src++;
23: #define COPY4()    COPY1(); COPY1(); COPY1(); COPY1();
24: #define COPY16()   COPY4(); COPY4(); COPY4(); COPY4();
25: #define COPY64()   COPY16(); COPY16(); COPY16(); COPY16();
26:
27: static inline void long_memcpy(long *dst, long *src, int count)
28: {
29:   short i = count / 64 - 1;
30:   do {
31:       COPY64()
32:   } while (--i != -1);
33: }
34:
35: void sub_read(void)
36: {
37:   int i;
38:   for (i = COUNT; i > 0; --i) {
39:       READ(fd, (UBYTE*)dst, SIZE);
40:   }
41: }
42:
43: void sub_read_bounce(void)
44: {
45:   int i;
46:   for (i = COUNT; i > 0; --i) {
47:       READ(fd, (UBYTE*)src, SIZE);
48:       long_memcpy((long*)dst, (long*)src, SIZE/sizeof(long));
49:   }
50: }
51:
52: double count(void (*p)(void))
53: {
54:   time_t t0, t1;
55:   int i, count;
56:   t0 = time(NULL);
57:   while (t0 == time(NULL))
58:       ;
59:   p();
60:   for (i = 0, t1 = time(NULL); t1 == time(NULL);i++)
61:       ;
62:   count = i;
63:   for (i = 0, t1 = time(NULL); t1 == time(NULL);i++)
64:       ;
65:   return (double)(t1-t0-1) - (double)(count) / (double)i;
66: }
```

```
67:
68: void main(int argc, char *argv[])
69: {
70:     double t;
71:     int sp;
72:     unsigned char *m = (unsigned char *)MALLOC(SIZE);
73:     sp = SUPER(0);
74:     goto dummy;
75: dummy:
76:     printf("%d KBytes Transfer %d times¥n", SIZE/1024, COUNT);
77:
78:     if (argc > 1 && (fd = OPEN((UBYTE*)argv[1], 0)) >= 0) {
79:         dst = m;
80:         t = count(sub_read);
81:         printf("HD   -> RAM      : %5.0lf KBytes/sec (%4.3lfs)¥n", (double
)(SIZE*COUNT)/t/1024.0, t);
82:         SEEK(fd, 0, 0);
83:         dst = (unsigned char*)(0xc00000);
84:         t = count(sub_read);
85:         printf("HD   -> GRAM     : %5.0lf KBytes/sec (%4.3lfs)¥n", (double
)(SIZE*COUNT)/t/1024.0, t);
86:         SEEK(fd, 0, 0);
87:         src = m;
88:         dst = (unsigned char*)(0xc00000);
89:         t = count(sub_read_bounce);
90:         printf("HD->RAM->GRAM    : %5.0lf KBytes/sec (%4.3lfs)¥n", (double
)(SIZE*COUNT)/t/1024.0, t);
91:         CLOSE(fd);
92:     } else {
93:         printf("bench <ファイル名>¥n");
94:     }
95:     SUPER(sp);
96: }
97:
```

リスト2 hdanim.c

```
  1: /*
  2:    HD アニメーションプログラム hdanim.c
  3:
  4:    gcc -O -fomit-frame-pointer -fstrength-reduce hdanim.c -ldos -liocs
-lfloatfnc
  5:
  6: */
  7: #include <stdio.h>
  8: #include <io.h>
  9: #include <stdlib.h>
 10: #include <string.h>
 11: #include <doslib.h>
 12: #include <iocslib.h>
 13: #include <ctype.h>
 14: #include <signal.h>
 15: #include <conio.h>
 16:
 17: #define TRUE        1
 18: #define FALSE       0
 19:
 20: #define CHRESO              '1'
 21: #define ESC                         0x1b
 22: #define ROLLUP              0x0a
 23: #define ROLLDOWN    0x0b
 24: #define STR_ROLLUP          "¥x0a"
 25: #define STR_ROLLDOWN        "¥x0b"
 26:
 27: #define SCREEN0     (0x18/2)
 28: #define SCREEN1     (0x1c/2)
 29: #define SCREEN2     (0x20/2)
 30: #define SCREEN3     (0x24/2)
 31:
 32: #define DISP_UPPER 0
 33: #define DISP_LOWER 1
 34: #define DISP_LEFT  0
 35: #define DISP_RIGHT 2
 36: #define DISP_ULMASK 1
 37:
 38: #define BUFFER_LARGE (512*256*sizeof(short)) /* 512dot x 256 line */
 39: #define BUFFER_SMALL (512*64*sizeof(short))  /* BUFFER_LARGE の1/4 */
 40:
 41: int totalframe=0;           /* 総表示枚数 */
 42: int totalfield=0;           /* 総フィールド数 */
 43: int faultfield=0;           /* 間に合わなかったフィールド数 */
 44:
 45: int crtmode = 14;           /* 画面モード */
 46: int speed = 4;              /* 一枚あたりの表示フィールド数 */
 47: int size = 256;             /* 入力の画面サイズ ( 256 / 128 ) */
 48: int zoomflag = FALSE;       /* 128x128 を拡大するかどうか */
 49: unsigned char *v_gram;      /* 読み込み用バッファ */
 50: int filesize;               /* ファイルサイズ */
 51: int fno;                    /* ファイルハンドル */
 52:
 53: volatile int skip = 1000;
 54: volatile int nowdisplay= DISP_UPPER | DISP_LEFT;   /* 現在の表示位置 */
 55: volatile int nowload = DISP_UPPER;          /* 現在読み込み中の領域 */
 56:
 57: char        backup[712];
 58:
 59: /*
 60:    表示位置変更
 61: */
 62: static inline void SetGraphicHome(int x, int y)
 63: {
 64:    volatile unsigned long *crtc = (volatile unsigned long *)0xe80000;
 65:    crtc[SCREEN0/2] = (x << 16) | y;
 66:    crtc[SCREEN1/2] = (x << 16) | y;
 67:    crtc[SCREEN2/2] = (x << 16) | y;
 68:    crtc[SCREEN3/2] = (x << 16) | y;
 69: }
 70:
 71: /*
 72:    垂直帰線割り込み処理
 73: */
 74: void        intterupt(void)
 75: {
 76:    int faultflag = FALSE;
 77:    __builtin_saveregs();
 78:    totalfield++;
 79:    if (--skip <= 0) {
 80:       switch (nowdisplay) {
 81:       case DISP_UPPER | DISP_LEFT:                    /* 左上を表示中 */
 82:          SetGraphicHome(256, 0);                      /* 右上に表示切り替え*/
 83:          nowdisplay = DISP_UPPER | DISP_RIGHT;
 84:          break;
 85:       case DISP_UPPER | DISP_RIGHT:                /* 右上を表示中*/
 86:          if (nowload == DISP_UPPER) {   /* 読み込みが終わっているか? */
 87:             SetGraphicHome(0, 256);               /* 左下に表示切り替え*/
 88:             nowdisplay = DISP_LOWER | DISP_LEFT;
 89:          } else {
 90:             faultflag = TRUE;
 91:          }
 92:          break;
 93:       case DISP_LOWER | DISP_LEFT:                 /* 左下を表示中*/
 94:          SetGraphicHome(256, 256);                /* 右下に表示切り替え*/
 95:          nowdisplay = DISP_LOWER | DISP_RIGHT;
 96:          break;
 97:       case DISP_LOWER | DISP_RIGHT:                /* 右下を表示中*/
 98:          if (nowload != DISP_UPPER) {   /* 読み込みが終わっているか? */
 99:             SetGraphicHome(0, 0);                 /* 左上に表示切り替え*/
100:             nowdisplay = DISP_UPPER | DISP_LEFT;
101:          } else {
102:             faultflag = TRUE;
103:          }
104:          break;
105:       }
106:       if (faultflag) {
107:          faultfield++;
108:       } else {
109:          skip = speed;
110:          totalframe++;
111:       }
112:
113:    }
114: }
115:
116:
117: /*
118:    終了処理部
119:    signal、atexit により呼ばれる
120: */
121: void        intterupt_end(void)
122: {
123:    VDISPST(NULL, 0, 0);
124:    CRTMOD(16);
125:    FNCKEYST(0,(unsigned char*)backup);
126:    fprintf(stderr, "¥x1b[>5l¥x1b*");
127:    while (kbhit()) {
128:       getch();
129:    }
130:    printf("総表示枚数:         %6d¥n", totalframe);
131:    printf("総表示フィールド:   %6d¥n", totalfield);
132:    printf("ドロップフィールド: %6d¥n", faultfield);
133:    printf("平均表示速度:       %6.3lf枚/秒¥n", (double)(totalframe) * 60.0 /
(double)(totalfield));
134: }
135:
136: /*
137:    キー入力処理
138: */
139: int         keycheck(void)
140: {
141:    int returnflag = TRUE, c;
142:    while (kbhit()) {
143:       returnflag = TRUE;
144:       switch(c = getch(), tolower(c)) {
145:       case ROLLUP :
146:          if (speed > 1) {
147:             speed --;
148:          }
149:          break;
150:       case ROLLDOWN :
151:          if (speed < 60-1) {
152:             speed++;
153:          }
154:          break;
155:       case CHRESO :
156:          crtmode = crtmode ^ 1;
157:          CRTMOD(0x100 | crtmode);
158:          break;
159:       case ESC :
160:          exit(1);
161:          break;
162:       default :
163:          returnflag = FALSE;
164:       }
165:    }
166:    return returnflag;
```

▶ AMIGAが生産再開！　よかった，よかった。　　鈴木　幹雄(35)神奈川県

```
167: }
168:
169: /*
170:    キー割り当て変更
171: */
172: void keyset(void)
173: {
174:    FNCKEYGT( 0,(unsigned char*) backup);
175:    FNCKEYST(21,(unsigned char*)STR_ROLLUP);              /*RollUp*/
176:    FNCKEYST(22,(unsigned char*)STR_ROLLDOWN);            /*RollDown*/
177:    fprintf(stderr, "¥x1b[>5h");
178: }
179:
180: /*
181:    使用法表示
182: */
183: void usage(void)
184: {
185:    fprintf(stderr,
186:           "¥n使用法¥thdanim <オプション> <画像データファイル>¥n"
187:           "オプション¥n"
188:           "        -l      15kHzモード¥n"
189:           "        -z      128x128を拡大表示する¥n"
190:           "        -s      128x128として再生する¥n"
191:           "        -t<n>   再生速度を n/60 秒にする デフォルト: 4 (秒15コマ)¥n"
192:           "キー操作¥n"
193:           "        ESC               終了¥n"
194:           "        'L'               15kHz/31kHz  切り替え¥n"
195:           "        ROLLUP            スピードアップ¥n"
196:           "        ROLLDOWN          スピードダウン¥n"     );
197:    exit(1);
198: }
199:
200: /*
201:    オプション解析
202: */
203: char *argcheck(int argc, char *argv[])
204: {
205:    char *picturefile = NULL;
206:    int i, tmp;
207:    for (i = 1; i < argc; ++i) {
208:       if (argv[i][0] == '-' || argv[i][0] == '/') {
209:          switch(toupper(argv[i][1])) {
210:          case 'H' :
211:          case '?' :
212:             usage();
213:             break;
214:          case 'L' :
215:             crtmode = 15;
216:             break;
217:          case 'S' :
218:             size = 128;
219:             break;
220:          case 'Z' :
221:             zoomflag = TRUE;
222:             break;
223:          case 'T' :
224:             if (atoi(&(argv[i][2])) != 0) {
225:                tmp = atoi(&(argv[i][2]));
226:                if (0 < tmp && tmp < 60) {
227:                   speed = tmp;
228:                }
229:             }
230:             break;
231:          }
232:       } else {
233:          picturefile = argv[i];
234:       }
235:    }
236:    return picturefile;
237: }
238: /*
239:    転送ルーチンの展開用マクロ
240:    X68030 だと、64個に展開するくらいが一番速い
241: */
242: #define COPY1()     *dst++ = *src++;
243: #define COPY4()     COPY1(); COPY1(); COPY1(); COPY1();
244: #define COPY16()    COPY4(); COPY4(); COPY4(); COPY4();
245: #define COPY32()    COPY16(); COPY16();
246: #define COPY64()    COPY32(); COPY32();
247: #define COPY256()   COPY64(); COPY64(); COPY64(); COPY64();
248:
249: /*
250:    GRAM 転送(256x256)
251: */
252: static inline void vram_copy_256(int dispmode)
253: {
254:    short i;
255:    unsigned long *dst;
256:    unsigned long *src = (unsigned long *)v_gram;
257:    if (dispmode == DISP_UPPER) {
258:       dst = (unsigned long *)(0xC00000);
259:    } else {
260:       dst = (unsigned long *)(0xC40000);
261:    }
262:    i = BUFFER_LARGE / sizeof(long) / 64 - 1;
263:    do {
264:       COPY64();             /* X68000だともっと多い方がいいかも */
265:    } while (--i != -1); /* こうすると dbra を使ってくれる */
266: }
267:
268: /*
269:    256x256 アニメーション処理
270:
271: /----- 512 dot -----¥
272: +----+----+----+----+
273: |         |         | ¥
274: |    1    |    2    |  > 256 line
275: |         |         | /
276: +----+----+----+----+
277: |         |         | ¥
278: |    3    |    4    |  > 256 line
279: |         |         | /
280: +----+----+----+----+
281: 1,2 の表示中に3,4を読み込む。
282: 3,4 の表示中に1,2を読み込む。
283: */
284: void anim_256(void)
285: {
286:    int filecount = filesize / (BUFFER_LARGE * 2);
287:    int i = filecount-1;
288:    /* 1,2 を読み込み、表示 */
289:    READ(fno, (UBYTE*)v_gram, BUFFER_LARGE);
290:    vram_copy_256(DISP_UPPER);
291:
292:    /* 3,4 を読み込み、表示 */
293:    READ(fno, (UBYTE*)v_gram, BUFFER_LARGE);
294:    vram_copy_256(DISP_LOWER);
295:    skip = speed;
296:    totalfield = 0;
297:    while(1) {
298:       /* 1,2 を読み込み */
299:       READ(fno, (UBYTE*)v_gram, BUFFER_LARGE);
300:       while ((nowdisplay & DISP_ULMASK) == DISP_UPPER)
301:          ;
302:       /* 1,2 をGRAMに転送 */
303:       vram_copy_256(DISP_UPPER);
304:       nowload = DISP_LOWER;
305:
306:       /* 3,4 を読み込み */
307:       READ(fno, (UBYTE*)v_gram, BUFFER_LARGE);
308:       if (KEYSNS()) {
309:          keycheck();
310:       }
311:       while ((nowdisplay & DISP_ULMASK) != DISP_UPPER)
312:          ;
313:       /* 3,4 をGRAMに転送 */
314:       vram_copy_256(DISP_LOWER);
315:       nowload = DISP_UPPER;
316:       if (--i == 0) {
317:          SEEK(fno, 0, 0);
318:          i = filecount;
319:       }
320:    }
321: }
322:
323:
324: /*
325:    GRAM 転送(128x128、拡大処理無し)
326: */
327: static inline void vram_copy_128_nozoom(unsigned short *dst, unsigned s
hort *src)
328: {
329:    short i, j;
330:    dst += 64 * 512 + 64;
331:    i = 127;
332:    do {
333:       j  = 128 / 64 - 1;
334:       do {
335:          COPY64();
336:       } while (--j != -1);
337:       dst += 128;
338:       j  = 128 / 64 - 1;
339:       do {
340:          COPY64();
341:       } while (--j != -1);
342:       dst += 128;
343:       src += 256;
344:    } while (--i != -1);
345: }
346:
347:
348: #define ZOOM1() col = *src++; *dst1++ = col;  *dst1++ = col;  *dst2++ =
col; *dst2++ = col;
349: #define ZOOM4()      ZOOM1(); ZOOM1(); ZOOM1(); ZOOM1();
350: #define ZOOM16()     ZOOM4(); ZOOM4(); ZOOM4(); ZOOM4();
351: #define ZOOM64()     ZOOM16(); ZOOM16(); ZOOM16(); ZOOM16();
352:
353: /*
354:    GRAM 転送(128x128、拡大処理有り)
355: */
356: static inline void vram_copy_128_zoom(unsigned short *dst, unsigned sho
rt *src)
357: {
358:    short i, j;
359:    unsigned short *dst1 = dst, *dst2 = dst + 512;
360:    unsigned short col;
361:    i = 127;
362:    do {
363:       j = 256 / 16 - 1;
364:       do {
365:          ZOOM16();
366:       } while (--j != -1);
367:       src += 256;
368:       dst1 += 512;
369:       dst2 += 512;
370:    } while (--i != -1);
371: }
372:
373: /*
374:    GRAM 転送(128x128)
375: */
376: void vram_copy_128(int dispmode)
377: {
378:    unsigned short *dst = (unsigned short *)(0xC00000);
379:    unsigned short *src = (unsigned short *)v_gram;
380:    switch (dispmode) {
381:    case DISP_UPPER | DISP_LEFT:
382:       break;
383:    case DISP_UPPER | DISP_RIGHT:
384:       dst += 512 * 256;
```

▶ああ，付録ディスクが増える。定期購読にしようかな……。でも，某所では前日に買えるしな。そう，これは定期購読にするとぼくらの掲示板に間に合わないという，とても困ったちゃんな悩みであります。　大古　哲生(23)千葉県

```
385:        src += 256;
386:        break;
387:    case DISP_LOWER | DISP_LEFT:
388:        src += 512 * 128;
389:        break;
390:    case DISP_LOWER | DISP_RIGHT:
391:        dst += 512 * 256;
392:        src += 512 * 128 + 256;
393:        break;
394:    }
395:    if (zoomflag) {
396:        vram_copy_128_zoom(dst, src);
397:    } else {
398:        vram_copy_128_nozoom(dst, src);
399:    }
400: }
401:
402:
403: /*
404:    128x128 アニメーション処理
405:
406: /--- 512 dot ---¥
407: +----+----+----+----+ ¥
408: |  1 |  2 |  3 |  4 | |
409: +----+----+----+----+ > 256 line
410: |  5 |  6 |  7 |  8 | |
411: +----+----+----+----+ /
412: 1-4 の表示中に5-8を読み込む。
413: 5-8 の表示中に1-4を読み込む。
414: */
415: void anim_128(void)
416: {
417:    int filecount = filesize / BUFFER_LARGE;
418:    int i = filecount-1;
419:    READ(fno, (UBYTE*)v_gram, BUFFER_LARGE);
420:    vram_copy_128(DISP_UPPER | DISP_LEFT);
421:    vram_copy_128(DISP_UPPER | DISP_RIGHT);
422:    skip = speed;
423:    totalfield = 0;
424:    while(1) {
425:        /* 1-4 の半分を読み込み */
426:        READ(fno, (UBYTE*)v_gram, BUFFER_LARGE/4);
427:        while ((nowdisplay & DISP_ULMASK) == DISP_UPPER)
428:            ;
429:        /* 5,6 をGRAMに転送 */
430:        vram_copy_128(DISP_LOWER | DISP_LEFT);
431:        nowload = DISP_LOWER;
432:        if (KEYSNS()) {
433:            keycheck();
434:        }
435:
436:        /* 1-4 の半分を読み込み */
437:        READ(fno, (UBYTE*)v_gram + BUFFER_LARGE/4, BUFFER_LARGE/4);
438:        while ((nowdisplay & DISP_ULMASK) != DISP_UPPER)
439:            ;
440:        /* 7,8 をGRAMに転送 */
441:        vram_copy_128(DISP_LOWER | DISP_RIGHT);
442:        nowload = DISP_UPPER;
443:        if (KEYSNS()) {
444:            keycheck();
445:        }
446:
447:        /* 5-8 の半分を読み込み */
448:        READ(fno, (UBYTE*)v_gram + BUFFER_LARGE/4 * 2, BUFFER_LARGE/4);
449:        while ((nowdisplay & DISP_ULMASK) == DISP_UPPER)
450:            ;
451:        /* 1,2 をGRAMに転送 */
452:        vram_copy_128(DISP_UPPER | DISP_LEFT);
453:        nowload = DISP_LOWER;
454:        if (KEYSNS()) {
455:            keycheck();
456:        }
457:
458:        /* 5-8 の半分を読み込み */
459:        READ(fno, (UBYTE*)v_gram + BUFFER_LARGE/4 * 3, BUFFER_LARGE/4);
460:        while ((nowdisplay & DISP_ULMASK) != DISP_UPPER)
461:            ;
462:        /* 3,4 をGRAMに転送 */
463:        vram_copy_128(DISP_UPPER | DISP_RIGHT);
464:        nowload = DISP_UPPER;
465:        if (KEYSNS()) {
466:            keycheck();
467:        }
468:
469:        if (--i == 0) {
470:            SEEK(fno, 0, 0);
471:            i = filecount;
472:        }
473:    }
474: }
475:
476:
477: void main(int argc, char *argv[])
478: {
479:    struct FILBUF buf;
480:    char *filename;
481:    int ssp;
482:
483:    if (filename = argcheck(argc, argv), filename == NULL) {
484:        usage();
485:    }
486:    if (v_gram = (unsigned char *)MALLOC(BUFFER_LARGE), (int)v_gram <0){
487:        printf("メモリを確保できません¥(%d kbyte)¥n",BUFFER_LARGE/1024);
488:        usage();
489:    }
490:    if ((fno = OPEN((UBYTE*)filename, 0)) < 0) {
491:        printf("ファイルが開けません(%s)¥n", filename);
492:        usage();
493:    }
494:    FILES(&buf, (UBYTE*)filename, 0x20);
495:    filesize = buf.filelen;
496:
497:    keyset();
498:    atexit(intterupt_end);
499:    signal(SIGINT, (void(*)(int))intterupt_end);
500:    CRTMOD(crtmode);
501:    G_CLR_ON();
502:    VDISPST((UBYTE*)intterupt, 0, 1);
503:
504:    ssp = SUPER(0);
505:    goto dummy_label;                    /* XC library の SUPER() のバグ回避 */
506: dummy_label:
507:
508:    if (size == 256) {
509:        anim_256();
510:    } else {
511:        anim_128();
512:    }
513:    SUPER(ssp);                                        /* 実際にここに来ることは無い*/
514:
515: }
516:
```

リスト3 gdump.c

```
 1: /*
 2:    hdanim用データ出力プログラム
 3:
 4:    gcc -O -fomit-frame-pointer -fstrength-reduce gdump.c -ldos -liocs -
lfloatfnc
 5:
 6: */
 7: #include <stdio.h>
 8: #include <stdlib.h>
 9: #include <string.h>
10: #include <doslib.h>
11: #include <iocslib.h>
12: #include <ctype.h>
13:
14: #define TRUE        1
15: #define FALSE       0
16:
17: #define BUFFER_SIZE (512*512*sizeof(short))
18:
19: int main(int argc, char *argv[])
20: {
21:    FILE *fp;
22:    int ssp;
23:    unsigned char *v_gram;
24:
25:    if (argc == 0) {
26:        printf("gdump <ファイル名>¥n");
27:        printf("¥tGRAM の内容を<ファイル> に追加します。¥n");
28:        exit(1);
29:    }
30:    if ((fp = fopen(argv[1], "ab")) == NULL) {
31:        printf("ファイル '%s' を開けません¥n", argv[1]);
32:    }
33:
34:    if (v_gram = (unsigned char *)MALLOC(BUFFER_SIZE), (int)v_gram <0) {
35:        printf("メモリを確保できません(%d kbyte)¥n", BUFFER_SIZE/1024);
36:        exit(1);
37:    }
38:
39:    ssp = SUPER(0);
40:    goto dummy;
41: dummy:
42:    memcpy(v_gram, (unsigned char *)(0xC00000), BUFFER_SIZE);
43:    SUPER(ssp);
44:
45:    fwrite(v_gram, 1, BUFFER_SIZE, fp);
46:    fclose(fp);
47:    return 0;
48: }
49:
```

各論2：Mach-2開発秘話
SCSI2ボードの可能性

Nakamura Chapuni 中村 巧

**X68000のディスク読み込み能力を4倍に拡大したSCSI2ボード
これがあって初めてHDアニメーション再生が可能になった
開発までの経緯とボードの特徴を見てみよう**

皆さん，はじめまして。今回SCSI2ボードMach-2の開発を担当しました，満開製作所の中村です。実は1日前にやっと出荷が始まったばかりで，それまでなにをしていたかといいますと,ROMを焼いていたり，マニュアルを書いていたりしたのです。マニュアルは出荷当日になってやっと完成したものらしいです。

8月号の瀧さんの紹介記事で,「メニューが起動し，いろいろなことができる(予定)らしい」と書かれていましたが，つまりその時点ではROMの内容はスカスカだったわけです。それでも，なんとか無事出荷を開始することができました。あとは無事に出荷を終えるだけです。

皆さんの温かいご声援に感謝しています！ そして,X68000PROユーザーの皆さん，「またもや」対応できなくて申し訳ありません。PROだと個体差により不安定になることが多いのと，RAMボードとの共存ができない，電源容量の不足でその他のボードとも共存が難しい，動いたときでも通常のSCSIボードの30%増しの速度がせいぜいであるなどの理由で対応することができませんでした。

さて，Mach-2の量産前に，試作基板を数枚ほど製造しました。何人かの方に貸し出して評価してもらったりしたのですが，なぜか，みーんな「アニメ」やってるんですよね。(U)さんも，DōGAのTさんも。満開でも，私が使用している試作基板のほかにもう1枚あり，それを(昇)氏に使ってもらっているのですが，いつの間にかハードディスクアニメーションに取り組んでいるようで，彼は「目指せ！ 動画でパズルだ！ ○ッ○ク○ー」とかいきまいております。もしかして，アニメやってないのは僕だけだったりするのでしょうか。

実際のところ，ハードディスクからの転送速度が向上することによって,やっと「まともな」ハードディスクアニメーションができるようになったといえるかもしれません。実はかくいう私も，Mach-2開発初期の頃は，ハードディスクの先頭セクタにグラフィック画面用の絵をベタ書きして，それをメインメモリとかGVRAMとの間で読み書きしてテストしていたものです。いつしかMach-2試作基板からHuman68kを起動してテストできるようになったので，ベタ絵を読み書きする機会はなくなってしまいました……。

ことの始まり

6月某日，浜町はOh!X編集部にMach-2試作基板を持っていったときの話です。編集部に置いてあったX68030に基板を挿して，dskbenchなどのベンチマークもそこそこに，(U)さんはどこかからMOを1枚持ってきて，ハードディスクにその内容をコピーし始めました。なにをするのだろうと私は眺めていただけでしたが，しばらくするとコマ送りのアニメーションが始まりました。本当にコマ送りと呼ぶにふさわしい感じでした。

「うーん。遅いなあ」

そのMOに入っていたものは，256×256ドット65536色の画像をベタで読み込んできてアニメーションさせるプログラム（編注：AMIです)，およびその画像データでしたが，たしかこのときは秒間3枚も映っていなかったような記憶があります。なるほど。MOと比べても遅い。ハードディスクから読み込んで秒間300Kバイトしか出ていない計算になってしまいます。いかん。このままでは「Mach-2は実は遅い」という悪評(?)が立ってしまう！

ここから今回のお話は始まります。

謎のノイズ

なぜ(U)さんの持ってきたアニメーションプログラムは遅かったのか……実は，このとき持ってきたMach-2はGVRAMへの直接読み込みを禁止していたのです。なぜかX68030では，Mach-2のバスマスタを用いてデータをGVRAMへ読み込むと，画面にゴミが出まくることがわかっていました。そのため，当時のバージョンのBIOSではGVRAMへの転送であった場合，ソフトウェア転送で処理するようになっていたのです。そして，(U)さんの持ってきたプログラムはGVRAMへの直接転送を行っていたのです。Mach-2のソフトウェア転送はあまりリキを入れて作っていなかったので,300Kバイト/秒というとても寒い転送速度しか出ていませんでした。

その場で私はMach-2のBIOSにパッチを当てて，GVRAMに対する転送であるかどうかのチェックを外しました。「用意できましたよ(U)さん。どーぞ心おきなく」結局，画面にバラバラとノイズをバラ撒きながらも秒間20フレームまで速度が上がり，(U)さんはけっこう満足げでした。

浜町からの帰路で私は考えていました。やっぱりGVRAMへの転送はどうにかしなくてはいけないのかなあ，と。

X68030で,GVRAMにバスマスタでデータを読み書きするとゴミが出る理由はいまだにわかっていません。ただ，どのようにバスタイミングをいじってみても，ゴミの出方はあまり変わらないようです。実は私もハードディスクアニメーションを狙っていて，なんとかしてGVRAMへの直接アクセスができるようにがんばっていたのですが，どうしてもゴミを出ないようにすることができなかったのでほとんど諦めていました。しかし,Oh!X編集部での一件で,GVRAMアクセスの大切さを再認識させられてしまいました。かといってこれは私の力量ではもうどうにも解決できない問題のようです。

とりあえず，この問題に対しては，ゴミ

を撒き散らしながらも強制的にGVRAMにバスマスタアクセスするIOCSコールを設けることによって対処しました。ゴミが出ても高速に読み書きできたほうがいい場合があると判断してのことです。IOCS $2Eがそれで，およそIOCS $26(READEXT)と同じパラメータが使用できます。

ただ，別コールにしてしまうと，Human68kでは使用できなくなってしまいます。そこで，SRAMの設定を変えることにより，とにかくGVRAMもバスマスタアクセスを行ってしまうような設定ができます。詳しくはMach-2のマニュアルをご覧いただくとしまして，$ED0098.Lのビット24を0にするだけです。Mach-2のインストールが無事に完了すると，この番地の内容はメモリ12Mバイトフル実装の場合，$FF000000になります。これを，$FE000000に変更するだけです。

気をつけなければいけないこととしましては，ディスクリフレッシャrefreshgでGVRAMをバッファに使用する設定にしていると，ディスクの内容が破壊されてしまうかもしれないということです。私もいっぺんハマってしまいました。

GVRAM転送の限界

GVRAMに直接読み込みを行った場合，メインメモリに読み込みを行う場合よりも多少速度が落ちてしまいます。I/Oスロットバスクロックが10MHz(100ns)のとき，メインメモリに対しては4クロック(400ns)でアクセスが完了するのに対して，GVRAMに対しては5クロック(500ns)かかってしまいます。

別の数値に換算すると，1回のアクセスで16ビットの転送ができますので，メインメモリ5Mバイト/秒に対してGVRAM 4Mバイト/秒です。実際にはハードディスクのセクタ間に休みが少し入ることと，DRAMリフレッシュのためにときどきウェイトが入ることにより，実効速度はもう少し下がります。だいたいそれぞれ4.3Mバイト/秒，3.3Mバイト/秒が限界値だと思われます。256×256, 65536色の画像は1フレームあたり128Kバイトですので，VRAMに直接読み込んだ場合の実質限界値は秒間26フレーム程度ということになります。

さて，GVRAMに直接読み込みを行うと，X68030の場合画面にゴミが乗ってしまいますが，ゴミが乗らないようにいったんメインメモリに読み込んできた場合はどのようになるでしょう。まず，SCSIのメインメモリへの読み込み速度が16ビットあたり400nsだとします。MC68030 25MHzがメインメモリをリードする場合，ノーウェイトで3クロックすなわち32ビットで120ns，GVRAMに書き込む場合11クロック，すなわち16ビットで440nsということになります。あわせて2.7Mバイト/秒，すなわち256×256，65536色の画像でかろうじて秒間20フレーム確保できるといったところでしょう。

風の便りによれば，(U)さんは結局いったんメインメモリを介して転送するようにしたらしいです。そして，秒間20フレームとか。なんだか先ほどの計算と一致していますね。

最後に

「とにかく高速なSCSIがほしい」一心で昨冬より研究開発をすすめてきたMach-2ですが，ようやく発売までこぎつけることができました。PROに対応できなかったことを始め，反省している点もたくさんありますが，とりあえずは4Mバイト/秒という目標を達成できたことで私自身はけっこう満足しています。ただ，X68000のI/Oスロットが10MHzで動いている関係で，理論上の転送速度限界値は5Mバイト/秒という感じです。(U)さんが先月のコラムで述べられていたように，5Mバイト/秒に大きなブレークスルーがあるとすれば，X68000においてまっとうな形状のボードではとうてい破れない壁ということになります。

MC68000のX68000なら5Mバイト/秒でとりあえず辛抱するとして，X68030で5Mバイト/秒に届かないというのはまだまだ十分ではないと個人的には思います。「あちらさん」の世界では，32ビットバスで10Mバイト/秒を軽く超え，むしろSCSIバスのほうが遅いといわれるまでになっています。現在私がX68030に接続して使用しているQuantum Atlas 1Gバイトも，Mach-2で4.3Mバイト/秒しか出ないのに，「あちらさん」の世界では8Mバイト/秒まで出るのです。

商品として発売するという前提があり，同時に商品として成り立たなければならないという制約もあったため，高速化のための大技としては「バスマスタ」を繰り出すところまでしかできませんでしたが，もし，個人的になにかを作るのであれば，「ローカルバス」にぜひとも挑戦してみたいところです。MPU直結のローカルバスとしましては，代表的なものとして040turboがありますね。Mach-2に触発されて誰かがローカルバスSCSIに挑戦することを私は密かに期待しています。そうすれば10Mバイト/秒も夢ではありません。

最後になりますが，話がX68030にかなり偏ってしまい，申し訳ありません。実際のところ，大量のデータを扱う段になるとX68000XVIでもまったくCPU速度が足りないところまできているのです。SCSIさえ速くなればなんとかなるのではないだろうかと私も思っていたのですが，今回Mach-2を製作して認識を改めてしまいました。

同期転送と非同期転送について

世の中の認識として，「SCSIの同期転送は速く，非同期転送は遅い」とよくいわれます。ところが，同期転送と非同期転送は世間でいわれているほど速度に違いはありません。これは私が実際にあれこれ試してみたので断言できます。

ただし，条件があります。

ホストアダプタ(イニシエータ)とデバイス(ターゲット)のケーブル長が十分に短いことです。ちょっと前の瀧さんの記事で同期転送と非同期転送について触れられていましたが，ケーブル長がある程度短ければ，非同期転送でネックとなっている「ケーブル中の信号伝搬速度」は表立って表れません。また，イニシエータとターゲットの，信号に対する反応速度も非同期転送のネックとなっていますが，実際にはFAST SCSIに対応しているようなSCSIコントローラであれば，反応速度はかなり速かったりします。結論からいってしまえば，3Mバイト/秒を超える転送速度を用いるのでなければ，非同期転送でも十分なのです。

実際の例を挙げておきましょう。

Quantum Empire 1080Sというハードディスクを接続していたとき，同期転送では3800Kバイト/秒の速度でした。非同期転送では，30cmのケーブルでいちばん近くに接続してもやはり3800Kバイト/秒の速度が出ていました。ところが，およそ3mくらい離して接続してみると，3200Kバイト/秒まで転送速度が落ちてしまいました。

さて，なぜ最後の最後で同期転送の話をしなければならなかったか説明しておきます。それは，ビデオ入力ユニットCZ-6VSIとMach-2を接続したとき，同期転送が設定されることがないからです。CZ-6VSIをMach-2で使用するときは，心持ちCZ-6VSIを近くになるように接続するのがよいでしょう。CZ-6VSIがハードディスクをアクセスするときは同期転送でアクセスを行うようですが，CZ-6VSIを本体からアクセスするときは，どうしても同期転送に設定できないのです。それはなぜかというと，とにかく，そういうものなのです。

[特集]

各論3：オフライン編集の基本
AMIデータ加工ツール

Kikuchi Isao 菊地 功

実はすでに応用性の高い無圧縮フォーマットを確立していたAMI
SCSI2ボードの能力によってAMIがようやく実用レベルになってきた
ここでは映像編集用の基礎ツールの作り方を紹介する

X680x0のアニメーション環境は，比較的初期の段階からある程度の道具は揃っていたものの，いまとなっては他機種に随分と水を明けられたかたちになってしまっています。その理由のひとつとして，メーカーからのサポートがまったくといっていいほどなかったということが挙げられます。かなり遅れてSX-WINDOWがCGAウィンドウをサポートしたものの，それほどパワフルなものでもなく，キャプチャリング装置であるビデオ入力ユニットが高価であることもあり，お世辞にもCGAウィンドウが広まっているとはいえません。

ここでいまさらメーカーに文句をいってもしかたありませんので，現状で考えてみましょう。やはりいまでももっとも一般的なのはDōGA CGAシステムではないかと思うのですが，ここでOh!X誌上で規定されたAMIに目を向けてみます。

AMIのメリットとしては，次のようなことが挙げられます。まず，ディスクからリアルタイムで読み込むので，メインメモリをほとんど必要としないこと。非圧縮なので，CPUパワーに依存しないこと。ライブラリが用意されているので，プログラムへの組み込みが容易なこと，などです。裏を返せば，膨大なディスク容量を必要とし，SCSI転送速度に依存するということになるのですが，MOやハードディスクは以前に比べればずいぶん安価になりましたし，満開製作所から高速SCSIボードも発売されますので，これはいまではそれほど致命的な欠陥ではないのではないでしょうか。

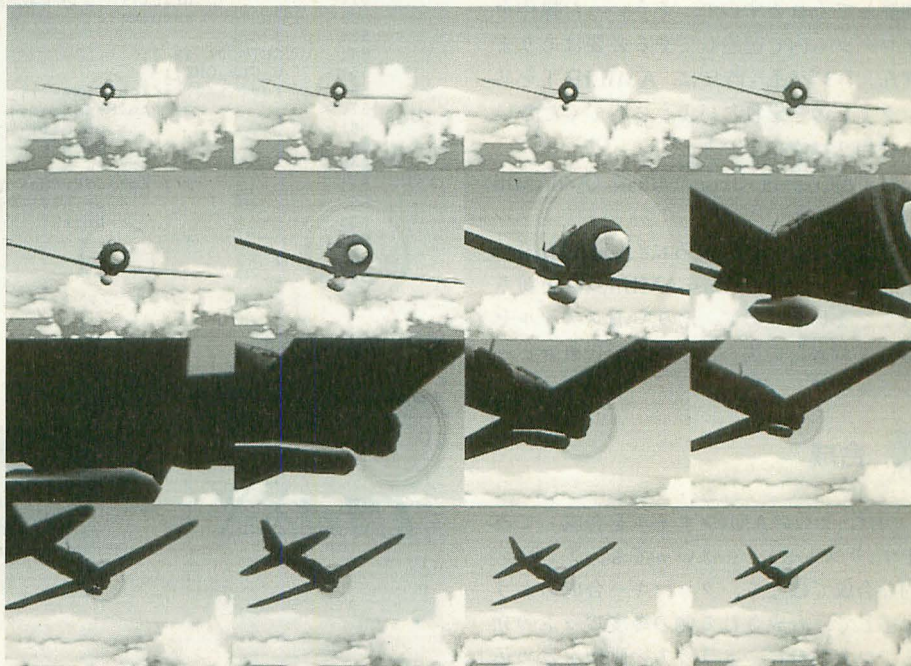
ただ，AMIの場合は(というか，たいていのアニメーションファイルは)，ひとつのファイルにまとまってしまっていますので，いったん作ってしまうとその編集はちょっと面倒です。もちろん編集ツールが充実していれば問題ないのですが，いまのところ，ごく基本的なコマンドと，決して高機能とはいえないEX-System用のAMI編集外部ファイルがあるだけです。それ以上のこととなると，自分でコマンドを作ったりしなければならなくなりますが，反対にいえば，AMIがライブラリを装備しているのは，「あとの環境は自分で作りなさい」ということなのかもしれません。

まあ，ある程度プログラムのできる人ならば，下手な編集ツールよりもライブラリを用意してくれたほうがありがたいでしょうが，だからといってライブラリだけぽんと渡されて「あとよろしく」っていわれても戸惑ってしまいますよね。そこで，今回は簡単なAMIの編集コマンドを紹介してみたいと思います。

AMI基礎知識

AMIには，「コマ」と「フレーム」という概念があります。一般的にはコマもフレームも同義語のように感じますが，AMIの場合は明確に区別されています。「コマ」というのは表示単位で，画面モードによってさまざまです。たとえば，128×128ドット65536色モードであれば，1コマは32Kバイトに相当します。それに対し，「フレーム」というのは，ディスクからの読み込み単位のことで，画面モードによらず1フレームは必ず512Kバイトです。したがって，上の例でいけば，1フレームは16コマということになります。では，コマはフレームの中でどのように格納されているのでしょうか。

結論から先にいってしまえば，図1のようになっています(128×128ドット65536色モード時)。これは，フレームデータをそのままG-RAM(512×512ドット65536色モード)に流し込んだイメージですが，この状態でコマは左上から右方向へ，右端まで達したら一段下の左端から右方向へ……とつながっています。この図から，まずフレー

図1 AMIデータの配列

ムを読み込んでおいて，ホーム位置を変えることでアニメーションさせているということは，もうお気づきですね（実際には半フレームずつ読み込むはず）。

さて，読み込みがフレーム単位ですので，AMIデータそのものもフレーム単位になっています(1024バイトのヘッダは別)。つまり，いままでの例でいえば，コマ数は16の倍数ということです。困ることもあるかもしれませんが，仕様ですので仕方ありませんね。

連結

まずは，2つのAMIファイルを連結してみましょう(リスト1)。片方から順にフレームを読み込んで，もう片方につなげていくという形式にします。このとき，画面モードのチェックは行っていますが，同期カウンタのチェックは行っていませんので，同期カウンタが異なっていても，連結される側の同期カウンタが優先されます。

フレームを読み込むには，AmiReadFrame( )関数を使用しますが，ここではG-RAMに流し込んでみます。こうしておけば，メモリの節約になるだけでなく，画像の確認にもなります(65536色モード以外では正しく表示されませんが，概観の確認くらいはできます)。

さて，次はフレームの書き込みです。任意のフレームに書き込む場合はAmiWriteFrame( )関数を使用しますが，フレームを付け足す場合にはAmiAddFrame( )関数を使用します。この関数はフレーム数も自動的に加算してくれますので，そういった操作をプログラム内で記述してやる必要はありません。コンパイル時には，AMILIB.Lをリンクするのを忘れないようにしてください。

使用法は次のとおりです。

AMIConnect <BaseAMIfile> <AddAMIfile>

<BaseAMIfile>の後ろに<AddAMIfile>を連結しますが，<BaseAMIfile>は更新されてしまいますので，元のAMIファイルをとっておきたい場合は，あらかじめコピーしておいてください。

合成

次は，2つのAMIファイルを合成してみます(リスト2)。とはいっても，アナログ的な合成ではなく，クロマキー合成のような感じで，合成される側の輝度ビットの立っている部分に，もう片方の画像をはめ込みます(図2)。その合成方法から，これは65536色モード専用ということにしておきましょう。

また，処理するフレーム数は，2つのAMIファイルの短いほうにあわせます。連結ではG-RAMを使いましたが，今度はテキストVRAMも使用し，合成する側をテキストVRAMに，される側をG-RAMにロードすることにしましょう。そうしておいて，G-RAMをスキャンし，輝度ビットが立っ

リスト1

```
 1: /*  AMI連結          (c)1995 Isawo-Kikuchi                */
 2: /*  AMIConnect <BaseAMIfile> <AddAMIfile>                  */
 3: /*  <BaseAMIfile>に<AddAMIfile>を連結する                 */
 4:
 5: #include        <stdlib.h>
 6: #include        <stdio.h>
 7: #include        <string.h>
 8: #include        <doslib.h>
 9: #include        <iocslib.h>
10: #include        <amilib.h>
11:
12: void            Connect( int, int );
13:
14: void            main( int ac, char *av[] )
15: {
16:     char        filename[90];
17:     int         fildes1, fildes2;
18:
19:     if( ac!=3 ){
20:                 printf( "AMIConnect <BaseAMIfile> <AddAMIfile>¥n" );
21:                 return;
22:     }
23:     strmfe( filename, av[1], "AMI" );
24:     fildes1 = AmiOpen( filename, 1 );
25:     if( fildes1<0 ){
26:                 printf( "%sがありません。¥n", filename );
27:                 return;
28:     }
29:     strmfe( filename, av[2], "AMI" );
30:     fildes2 = AmiOpen( filename, 0 );
31:     if( fildes2<0 ){
32:                 printf( "%sがありません。¥n", filename );
33:                 CLOSE( fildes1 );
34:                 return;
35:     }
36:     Connect( fildes1, fildes2 );
37:     CLOSE( fildes1 );
38:     CLOSE( fildes2 );
39: }
40:
41: void            Connect( int fildes1, int fildes2 )
42: {
43:     int         i, fcnt;
44:
45:     if( AmiGetMode( fildes1 )!=AmiGetMode( fildes2 ) ){
46:                 printf( "再生モードが異なります。¥n" );
47:                 return;
48:     }
49:     fcnt = AmiGetFcnt( fildes2 );
50:     if( fcnt<0 ){
51:                 printf( "フレーム数が異常です。¥n" );
52:                 return;
53:     }
54:     CRTMOD( 12 );
55:     G_CLR_ON();
56:     for( i=0; i<fcnt; i++ ){
57:                 if( AmiReadFrame( fildes2, 0xC00000, i )<0 ){
58:                         printf( "ファイルが異常です。¥n" );
59:                         break;
60:                 }
61:                 if( AmiAddFrame( fildes1, 0xC00000 )<0 ){
62:                         printf( "ディスクフルです。¥n" );
63:                         break;
64:                 }
65:     }
66: }
```

図2　合成のようす

ているピクセルだけをテキストVRAMからコピーし，今度はAmiWriteFrame( )関数で再びG-RAMを書き戻してやればいいわけです。ただし，プログラムでは，flagという更新フラグを使用して，余計なリード/ライトをしないようにしてあります。使用法は次のとおりです。

AMICompose 〈BaseAMIfile〉〈AddAMIfile〉

〈BaseAMIfile〉の輝度ビットが立っている部分に〈AddAMIfile〉がはめ込まれますが，連結と同様に〈BaseAMIfile〉は更新されますので，気をつけてください。

AMIの展望

今回の2つのツールが実用的かどうかは別にして，よっぽど変なことをするのでなければ，AMIファイルをいじるのはそんなに難しくないということがわかっていただけたかと思います。

たとえば合成するにしても，まず輝度ビットを立てるという作業は必要になってきますよね。そんなときは1コマずつ書き込むよりも，フレーム単位でG-RAMに流し込んでからグラフィックエディタで書き込んで，G-RAMをフレームに書き戻すといったほうが効率的です。

こういったちょっとしたツールの積み重ねて，AMIの環境はどんどん充実したものになっていくのではないでしょうか。皆さんも「こんなことしたいな」と思ったら，「でも，めんどくさそう」などと思わずに，挑戦してみてください（AMIに限った話ではありませんよ）。

新しい映像編集環境

デジタルでのアニメーション編集を効率よく行うためには，こういったフィルタを使いやすくまとめることが必要になってきます。しかし，HDアニメーションを再生できる基礎環境ができてしまえば，ユーザーインタフェイスを作ること自体はさほど難しいことではないでしょう。

AMIならファイルフォーマットも決まっていますし，データは無圧縮，均等サイズとなればフェーダやワイパーはすぐに作れます。今回はAMI以外のフォーマットも2つ示唆されていますが，基本的な考え方にさほど大きな違いはありません。編集のしやすさを考慮すれば，G-RAM構造から離れたUSAGI環境に移行していくべきでしょう。それは220×220ドット32768色，無圧縮データの単純な結合体。ヘッダは不要。輝度ビットは編集時にマスクとして使用されます。各画面の第1ラスタに相当する440バイトは拡張用で使用不可となります。再生レートは20fpsと15fpsのみ想定し，再生環境とプレイヤー側で選択します。

リスト2

```
==============  amicompose.c  =================
  1: /*  AMI合成          (c)1995 Isawo-Kikuchi                */
  2: /*  AMICompose <BaseAMIfile> <AddAMIfile>                  */
  3: /*  <BaseAMIfile>の輝度ビットが立っている領域に      */
  4: /*  <AddAMIfile>を合成する                               */
  5:
  6: #include      <stdlib.h>
  7: #include      <stdio.h>
  8: #include      <string.h>
  9: #include      <doslib.h>
 10: #include      <iocslib.h>
 11: #include      <amilib.h>
 12:
 13: void          Compose( int, int );
 14:
 15: void          main( int ac, char *av[] )
 16: {
 17:       char    filename[90];
 18:       int     fildes1, fildes2;
 19:
 20:       if( ac!=3 ){
 21:               printf( "AMIConnect <BaseAMIfile> <AddAMIfile>¥n" );
 22:               return;
 23:       }
 24:       strmfe( filename, av[1], "AMI" );
 25:       fildes1 = AmiOpen( filename, 1 );
 26:       if( fildes1<0 ){
 27:               printf( "%sがありません。¥n", filename );
 28:               return;
 29:       }
 30:       strmfe( filename, av[2], "AMI" );
 31:       fildes2 = AmiOpen( filename, 0 );
 32:       if( fildes2<0 ){
 33:               printf( "%sがありません。¥n", filename );
 34:               CLOSE( fildes1 );
 35:               return;
 36:       }
 37:       Compose( fildes1, fildes2 );
 38:       CLOSE( fildes1 );
 39:       CLOSE( fildes2 );
 40: }
 41:
 42: void          Compose( int fildes1, int fildes2 )
 43: {
 44:       int     i, j, fcnt, ssp, flag;
 45:       unsigned short  *crtc, *gp, *vp;
 46:
 47:       i = AmiGetMode( fildes1 );
 48:       if( (i>>2)!=4 ){
 49:               printf( "65536色モードではありません。¥n" );
 50:               return;
 51:       }
 52:       if( i!=AmiGetMode( fildes2 ) ){
 53:               printf( "再生モードが異なります。¥n" );
 54:               return;
 55:       }
 56:       fcnt = AmiGetFcnt( fildes1 );
 57:       if( fcnt<0 ){
 58:               printf( "フレーム数が異常です。¥n" );
 59:               return;
 60:       }
 61:       i = AmiGetFcnt( fildes2 );
 62:       if( i<0 ){
 63:               printf( "フレーム数が異常です。¥n" );
 64:               return;
 65:       }
 66:       if( i<fcnt ) fcnt = i;
 67:       CRTMOD( 12 );
 68:       G_CLR_ON();
 69:       OS_CUROF();
 70:       ssp = SUPER( 0 );
 71:       crtc = (unsigned short *)0xE82600;
 72:       *crtc &= 0xFFDF;                            /* テキスト非表示 */
 73:       gp = (unsigned short *)0xC00000;            /* グラフィックVRAM */
 74:       vp = (unsigned short *)0xE00000;            /* テキストVRAM */
 75:       for( i=0; i<fcnt; i++ ){
 76:               if( AmiReadFrame( fildes1, gp, i )<0 ) break;
 77:               for( j=flag=0; j<512*512; j++ ){
 78:                       if( gp[j]&1 ){              /* Base側の輝度ビット */
 79:                               if( !flag ){        /* 更新フラグ */
 80:                                       if( AmiReadFrame( fildes2, vp, i )<0 ) g
oto quit;
 81:                                       flag = 1;
 82:                               }
 83:                               gp[j] = vp[j];
 84:                       }
 85:               }
 86:               if( flag ) if( AmiWriteFrame( fildes1, gp, i )<0 ) break;
 87:       }
 88: quit:
 89:       for( j=0; j<512*512; j++ ) vp[j] = 0;       /* テキストVRAMクリア */
 90:       *crtc |= 0x20;                              /* テキスト表示 */
 91:       SUPER( ssp );
 92:       OS_CURON();
 93:       if( i<fcnt ) printf( "エラーが発生しました。¥n" );
 94: }
```

[特集]

Animation Now!

各論4：最近の圧縮技法を探る
シネパックのアルゴリズムを見る

Kikuchi Isao 菊地 功

他機種では標準的に使われているシネパック
軽い動作とそこそこの画質，高い圧縮率が魅力のアルゴリズムだ
ここではその符号化の詳細について見てみよう

先月の付録ディスクに掲載されたシネパックプレイヤー（以降単にシネパックプレイヤーと記す），楽しんでいただけているでしょうか？　友人からは，「危険なツール」の称号をいただきました。このツールによって，「先にエンディングを見たらゲームをする気がなくなった」「クソゲーだと思いつつも，アニメーションが多そうなので買ってしまった」「サターンも持っていないのに，ソフトを集めてしまった」といった苦情は受け付けておりませんので，ご了承ください（先日U氏＆J氏とバ○ボンズを見て唖然としてしまった）。

先月も簡単にシネパックについての紹介をしましたが，今月はそのフォーマットについて詳しく説明していきましょう（自己解析ですが）。

シネパックについて

先月もいいましたが，シネパックとは，Super Mac社が開発した動画再生フォーマットのひとつで，QuickTimeやVideo for Windowsのフォーマットとしても採用されています。圧縮率が高く，デコードが軽いという特徴から，パソコンなどのレベルではかなり有効なフォーマットのようです。

一般的には，圧縮率とデコード時間は反比例すると考えていいでしょう。非圧縮であればデコード時間が限りなく0に近くなります。これをやっているのがAMIですね。ただし，こういった動画再生はたいていの場合，ハードディスクなどの外部記憶装置からリアルタイムに再生するものです。しかし，以前よりも随分速くなったとはいえ，ハードディスクはCPUから見ればまだかなり遅いデバイスと考えられます。そこで，ある程度CPUパワーのあるマシンでは，圧縮をかけてディスクからの読み込み量を減らし，ソフトウェアで伸長することで，全体としての速度を上げようという考えに行き着きます。シネパックは，この「圧縮率」と「伸長時間」のバランスが取れたフォーマットといえるでしょう。

サターンのブルーシード

しかし，シネパックに限らず，こういった圧縮を行う動画再生フォーマットのほとんどは可逆ではありません。静止画をJPEGでセーブした場合と同じようなものです。しかし，動画では再生スピードを稼ぐことと，「動いていれば人間の目はごまかせる」という考えから，画質はJPEGと比べるとかなり悪くなるのが一般的です（専用のハードを積んでいれば別ですが）。また，エンコード時に画質をわざと落として，再生速度を稼ぐという手段もあります（画質を落とせば情報量は減らせますから，圧縮率はよくなります）。シネパックもその例に漏れず，残念ながら画質は決していいとはいえません（個人的にはシネパックよりもIndeo3.1のほうが好きです）。

さて，シネパックとひと言でいっても，ファイルフォーマットのなにからなにまで決められているわけではありません。あくまで連続した画像の圧縮フォーマットであって，その他のヘッダ，タイムテーブル，音声フォーマットなどについては特に規定されていないようです。たとえば，Video for Windowsのデータでは，シネパックはあくまでもフォーマットのひとつにすぎませんから，ヘッダには「これはVideo for Windowsのデータだよ」という識別子が入っていますし，シネパックとは直接関係のないデータが入っていたりもするようです。また，3DO（こちらも動画再生の標準はシネパック，ただし普通の方法ではCD-ROMを見ることはできない）では，フォーマットが何種類かあるようです。こうなってくると，かなり骨が折れるうえに，すべてのフォーマットをサポートするというのは困難になってきます。では，サターンはというと，幸運なことに，いまのところはフォーマットは厳密に規定されているようです（将来的にデコーダあるいはエンコーダがバージョンアップした場合はどうかわからないが）。しかも，比較的素直なフォーマットで，音声（PCM）データも非圧縮です。というわけで，以降はサターンのシネパックに添って解説していくことにします（単にシネパックといった場合には，サターンのシネパックを指すことにします）。

シネパックの構造

シネパックファイルは，大きく4つのパートに分けることができます。

1)　ヘッダ

ファイルの先頭にあり，表1のような情報が含まれています。'FILM'という識別子は，もちろんこのファイルが動画データであることを，'FDSC'はフォーマットを，'cvid'はシネパックであることを示しているのでしょう。'FDSC'や'cvid'という文字列にはそれぞれ意味があるのでしょうが，申し訳ありませんが勉強不足で私には詳しくはわかりません。いまのところは固定と思って差し支えないでしょう。

バージョンというのはおそらくエンコーダのバージョンだと思われます。必ずしも固定ではないのですが，シネパックプレイヤーでは無視しています。その他の情報については特に問題ないでしょう。(?)マークがついている部分は，すべてのファイルで共通だった部分です。あまり考えないこと

にしましょう。

2) タイムテーブル

ヘッダの直後，アドレスで$30の位置にあり，タイムテーブルの内部はさらにヘッダと複数のレコードからなっています（表2）。レコードにはPCMレコードと画像レコードがあり，それぞれPCM/画像データへのアドレスなどを示しています（順番はこの限りではありません）。ただし，ここでの先頭アドレスとは，タイムテーブル直後をオフセットとしたアドレスで，この例でいえば$1E40を加えたアドレスがファイル先頭からの絶対アドレスになります。画像レコード内の表示開始カウンタ，表示時間は，共にタイムテーブルのヘッダ中のタイムカウンタを基準とした時間で表されています。この例では表示時間は$28＝40ですから，40×1/600＝1/15で，秒間15コマのデータということになります。

もう一度$60からのレコードでやってみましょう。オフセットからのアドレス$7F68，つまり絶対アドレス$9DA8から$30FCバイトの画像データは，表示開始カウンタ$28，つまり再生開始1/15秒後から1/15秒間表示される，ということになりますね。と，ここで表示開始カウンタの最上位ビットが立っているのに気がつきます。おそらく再生が間にあわなかったときの動作を示すフラグの類だと思うのですが，気にしないことにします。

ところで，画像データは1コマごとにたくさんあってもいいとして，なぜPCMデータが複数あるのでしょう。しかも画像データとPCMデータがお互いにサンドイッチされたかたちで。答えは，これも「リアルタイム」ということに関係してきます。動画再生はほとんどの場合，リアルタイムであるということは先ほど述べました。これは，映像はもちろん，音声もということです。すると，ある一瞬を考えたときに，映像データと音声データを同時に読み込まなくてはなりません。しかし，実際問題としてそれは不可能なので，音声データを細切れにして，映像データの間に挟むということになります。ある一定の時間分のPCMデータを読んでおいて，そのPCMが鳴っている間に映像データを読み込んで表示する，あるいはその反対といった感じですね（だいたい画像4～5コマにつきPCM1ブロック）。

では，映像データと音声データを別々に用意しておいたらどうなるでしょう。それぞれにファイルポインタを当てて，ちょっとずつつまんでいけば，それでいいような気もします。しかし，ソフトウェア的にはそれぞれを連続して読んでいるように見えても，物理的には（同一ドライブである限り）ヘッドのシークを避けられません。このシークという動作は，ただでさえ遅い外部記憶装置のなかでももっとも遅い部類の動作に入ります。

結局，サンドイッチにするのが速度的にもっとも有利ということになります。ただし，これはあくまでも音声データもリアルタイムである場合の話です。X680x0では最高でも15.6kHz，しかもAD PCMですので，データ量はたかが知れています。そこで，シネパックプレイヤーでは，まずはPCMデータを逐次読み込み周波数AD PCM変換してメモリに蓄え，せーのでPCMはメモリから，画像はディスクから読むようにしています。多少のメモリは必要としますし，最初にPCMを読む時間はかかりますが，再生中は画像側に専念できますので，こうしたほうが速くなるのは当然のことですね。

3) 音声データ

タイムテーブルで示されたアドレスから，ヘッダで示されたPCMがべたで格納されています。8ビット，16ビットとも，最上位ビットが符号ビットの最も一般的な形式ですが，ステレオの場合はちょっと注意が必要です。というのは，RとLが完全に分離されているからです。たとえば，データがn個あったとすると，Rはn-1個目まで，Lがn個目からといった具合です（RとLは反対かも）。シネパックプレイヤーでは，ステレオの場合は，左右を合成（平均）するようになっています。

さて，こういったPCMデータをX680x0で鳴らすには，まず周波数変換を行ってから，PCM→ADPCM変換をする必要があります。AD PCM変換については，本誌1992年6月号のPCM8の記事に詳しく説明してありますのでそちらを参照してもらうことにして，周波数変換について簡単に説明しておきましょう。

いま44.1kHzのデータを22.05kHzに変換するとしましょう。これは大して考える間もなく，データを1個飛ばしで拾っていけばいいということはわかりますね。44.1:22.05＝2:1ですから，2個のデータのうち1個拾えばいいわけです。じゃあ，22.05kHzを15.6kHzにするには？　え～っと，22.05:15.6≒17:12だから……なんていちいちやっていたら大変ですし，どんどん誤差が出てきてしまいますよね。そこで，ちょっと考え方を変えます。

ある時間原点を考え，そこから1/22050秒ごとに目盛りを振っていけば，その1目盛りが22.05kHzのデータ1個分になります（図1上）。それに対し，1/15600秒ごとに目盛りを振れば，1目盛りは15.6kHzのデー

表1 ヘッダ

```
ヘッダ（48バイト）

00000000  46 49 4C 4D          識別子(FILM)
00000004  00 00 1E 40          ヘッダ+タイムテーブルのサイズ
00000008  31 2E 30 34          バージョン(1.04)
0000000C  00 00 00 00          固定(?)
00000010  46 44 53 43          識別子(FDSC)
00000014  00 00 00 20          FDSCブロックのサイズ（含む識別子）
00000018  63 76 69 64          識別子(cvid)
0000001C  00 00 00 E0          画像Yサイズ
00000020  00 00 01 40          画像Xサイズ
00000024  18               画像ビット数(?)
00000025  01               PCM 01..モノラル 02..ステレオ
00000026  08               PCM 08..8ビット 10..16ビット
00000027  00               固定(?)
00000028  56 22                PCM 周波数(Hz)
0000002C  00 00 00 00 00 00    固定(?)
```

表2 タイムテーブル

```
ヘッダ

00000030  53 54 41 42          識別子(STAB)
00000034  00 00 1E 10          STABブロックのサイズ（含む識別子）
00000038  00 00 02 58          タイムカウンタ(1/600秒)
0000003C  00 00 01 E0          レコード数

PCMレコード（16バイト）

00000040  00 00 00 00          先頭アドレス
00000044  00 00 2B 10          データ長
00000048  FF FF FF FF          PCM識別子
0000004C  00 00 00 01          固定(?)

画像レコード（16バイト）

00000050  00 00 2B 10          先頭アドレス
00000054  00 00 54 58          データ長
00000058  00 00 00 00          表示開始カウンタ（画像識別子）
0000005C  00 00 00 28          表示時間

以降同様にレコードが続く

00000060  00 00 7F 68 00 00 30 FC 80 00 00 28 00 00 00 28
00000070  00 00 B0 64 00 00 15 88 FF FF FF FF 00 00 00 01
    :
00001E30  00 49 2A 5C 00 00 26 7C 80 00 3B 10 00 00 00 28
```

タ1個分です(図1下)。したがって，下のグラフのそれぞれの目盛りが，上のグラフの目盛りのどの位置にくるかを調べ，その位置のデータを引っ張ってくれば，22.05kHzのデータを15.6kHzに変換できることになります。

ここで，上と下のグラフの一目盛りの長さの比は15600:22050ですから，ある瞬間の目盛りのずれをLとすると，L＝L－15600でL<0となるときのデータを拾ってL＝L＋22050する動作を繰り返せば，精度よく周波数を落とすことができます。反対に11.025kHzを15.6kHzに周波数を上げる場合では，同様にL＝L＋11025でL≧0となるまで同じデータを拾い，L＝L－15600すればいいことになります。これらの2つをまとめてC言語で書くと，リスト1のようにたったこれだけの処理で済んでしまいます。ちょっとわかりにくいかもしれませんが，処理を追って，上のような流れになっていることを確認してください。

ちなみにシネパックプレイヤーではこの部分はアセンブラで書いているのですが，Lに相当するものは16ビットで演算していますので，周波数は65535Hzまでにしか対応していません。普通に使っている限り問題になるとは思えませんが。

4)　画像データ

シネパックは，2×2のパターンと4×4のパターンを，それぞれ最大256個ずつ登録しておいて，それを並べて画像を作るという方式が基本です。パターンおよび画像は各コマごとに差分更新されるのですが，画像が大きな場合は上下に2分割して管理し，それぞれ別のパターンを登録することが許されているようです（ひょっとしたら3分割以上も許されているのかもしれないが，シネパックプレイヤーは2分割までに対応)。また，3DOのシネパックの一部には，2コマ前との差分で保存されているものもありますが，とりあえず考えないことにします。

画像データは，さらにヘッダとデータ部に分けられます（表3)。最初の2バイトは，最初のコマだけが$0000で，あとのコマはすべて$0001のようですので，それらの識別にでも使われるのでしょう。とりあえずあまり考えなくてもいいようです。次の2バイトはブロックの長さを示しているのですが，実際よりも8バイト小さいというのは，画面分割数以降のバイト数を示しているのでしょうか。画面分割数は，先ほどいった上下に分割する数ですね。例では，2分割されています。データ部は，2分割されている場合は上画像と下画像に分けられ，それぞれがさらにヘッダとシネパックの生データ部に分けられます。ここのヘッダの最初の2バイトも，最初だけが$1000で，それ以降(最初のコマの下画像データも)$1100のようですので，これも気にしないことにします。

ここで，ヘッダのXYサイズと，上画像データのXYサイズを見比べてみましょう。Yサイズだけが半分になっていますね。これは上画像データでちょうど上半分だけを描画できることを示しています。

さて，いよいよシネパックそのもののデータ構造に入ります。この部分はサターンに限らず，どんなシネパックでも共通であると思われます。つまり，シネパックCODECが厳密に規定されている部分ですね。この生データ部分も，識別子によって3種類のパートに分類されています。

まずは識別子が$2000あるいは$2100の場合です(表4)。これは2×2パターン登録用のデータで，一番最初にどちらか一方だけがきます。どちらの場合もYが4個，

表3　1コマ目の構造

画像データ（1コマ目）

ヘッダ		
00004950	00 00	一番最初のコマの識別(?)
00004952	54 50	このブロックの長さ-8
00004954	01 40	Xサイズ
00004956	00 E0	Yサイズ
00004958	00 02	画面分割数(?)
0000495A	00 00	固定(?)
上画像データ		
0000495C	10 00	一番最初の映像データ識別(?)
0000495E	2A 2C	このブロックの長さ
00004960	00 00 00 00	固定(?)
00004964	00 70	Yサイズ
00004966	01 40	Xサイズ
:	以降シネパック生データ	
下画像データ		
00007388	11 00	一番最初の映像データ識別(?)
0000738A	2A 20	このブロックの長さ
0000738C	00 00 00 00	固定(?)
00007390	00 70	Yサイズ
00007392	01 40	Xサイズ
:	以降シネパック生データ	

表4　シネパック生データ

2×2パターン登録

00004968	20 00	2×2パターン登録識別子
0000496A	06 00	このブロックの長さ
0000496C	73 77 74 74	2×2パターンNo.0の Y1 Y2 Y3 Y4
00004970	E4 18	2×2パターンNo.0の Cb Cr
00004972	B0 B4 B3 B3	2×2パターンNo.1の Y1 Y2 Y3 Y4
00004974	ED 0E	2×2パターンNo.1の Cb Cr
:		
00007394	21 00	2×2パターン登録識別子
00007396	05 F4	このブロックの長さ
00007398	FF FF FF FF	パターン登録フラグ
0000739A	76 76 76 76	2×2パターンNo.0の Y1 Y2 Y3 Y4
000073A0	E2 1B	2×2パターンNo.0の Cb Cr
:		

図1　周波数変換(高→低)

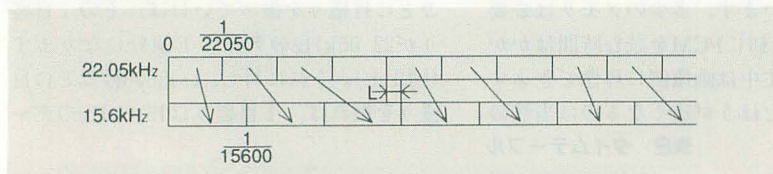
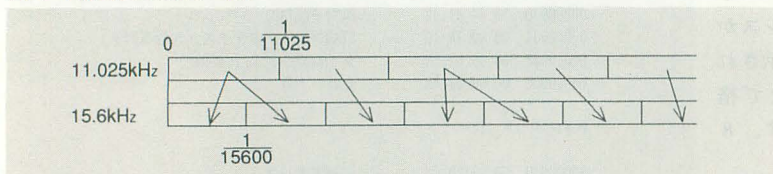



図2　周波数変換(低→高)

リスト1

```
 1:
 2:     int     freq1;  /* 変換前周波数 */
 3:     char    *pcm1;  /* 変換前データ格納領域 */
 4:     int     n;      /* 変換前データ個数 */
 5:     int     freq2;  /* 変換後周波数 */
 6:     char    *pcm2;  /* 変換後データ格納領域 */
 7:     int     i, L;
 8:     char    dat;
 9:
10:     for( i=L=0; i<n; i++ ){
11:             while( L>=0 ){
12:                     dat = *(pcm1++);
13:                     L -= freq2;
14:             }
15:             while( L<0 ){
16:                     *(pcm2++) = dat;
17:                     L += freq1;
18:             }
19:     }
```

Cb，Crが1個ずつで，図3のようなパターンを構成します。ただし，ここで示した式はRGBそれぞれ8ビットのフルカラーとして計算していますので，65536色で表すためにはRGBをそれぞれ3ビットシフトダウンしなければなりません。

2つの違いは，先頭から順に登録していくか，あるいは登録不要の部分をスキップ(差分登録)するかどうかです。$2000の場合は，パターンNo.0から順にYCCが規則正しく並んでいます。ただし，No.255まであるかどうかはわかりませんので，個数はブロック長から判断します。それに対し，$2100はまず32ビットのフラグを拾い，その上位から1ビットを見て，パターンを更新するかどうかを判断します。

例ではすべてのビットが立っていますので，少なくとも最初の32パターンは更新しますが，たとえば$0000FFFFだった場合，先頭のNo.15まではスキップし，No.16から16個を更新することになります。もちろんファイルには更新される部分のデータしか保存されていません。こうしてパターンを登録していき，フラグを32ビットすべて使いきったところで，再び4バイト拾ってフラグとします。これをこのブロックが終わるまで繰り返すことになります。C言語で

図3　2×2パターンの構成

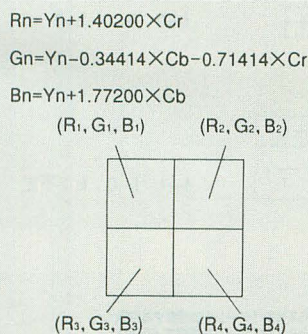



表5　シネパック生データ(画像データ本体)

0000798C	30 00	画像本体識別子
0000798E	24 1C	このブロックの長さ
00007990	BF FC F6 FF	パターン展開フラグ
00007994	7A 33 76 A5	2×2パターン番号×4
00007998	27	4×4パターン番号
:		
0000A3DC	31 00	画像本体識別子
0000A3DE	12 14	このブロックの長さ
0000A3E0	D0 00 00 06	パターン展開フラグ
0000A3E4	49 6F 74 70	2×2パターン番号×4
0000A3E8	3C	4×4パターン番号
:		
0000E284	32 00	画像本体識別子
0000E286	11 84	このブロックの長さ
0000E288	54	4×4パターン番号
:		

書くと，リスト2のようになります(これはわかりやすくするためのプログラムで，実用向けではない)。

これらの性質から考えて，$2000は一番最初だけに，$2100はそれ以降ということになるようです。また，画面が分割されていた場合，$2000はもう一方のパターンにも影響するようです。

次にくるのが，識別子$2200あるいは$2300です。こちらは4×4パターン登録用のデータで，それぞれ$2000と$2200，$2100と$2300がまったく同じデータ構造をしています。違うのは，実際に描画される際に縦横2倍に引き伸ばされるということです。つまり2×2の同色の四角が2×2個並ぶことになります。画面が分割されていた場合の$2200も，もう一方のパターンに影響を与えます。

最後は識別子$3000，$3100あるいは$3200の本体部分です(表5，$3300もあるようなのですが，私が調べた限りでは出てきませんでした)。すでに登録されている2×2および4×4パターンを並べて画像を作ります。これらはちょっとヘビーなので，1つずつ順を追って説明していきましょう。

まずは描画の単位です。2×2パターンは単独ではいうまでもなく2×2ドットですが，必ず4個セットで左上，右上，左下，右下の順で現れます。4×4パターンは基本的に単独で現れます。また，差分更新ですから，描画をスキップすることもあるのですが，これは必ず4×4ドット単位でスキップされます。したがって，描画の単位(スキップすることも描画の一種と考える)は4×4ドットということになります。

さて，まずは$3000ですが，$2100などと同じように，32ビットのフラグを使います。ただし，今度はフラグが立っているときは2×2のパターンを4個，立っていないときには4×4のパターンを描画するようにします。C言語で書くと，*リスト3*のようになります(これも実用向けではない)。これからわかるように，描画をスキップすることはしません。したがって，$3000は主に一番最初のコマで使われます。

次は$3200を先に行きましょう。これは滅多に出てこないのですが，フラグもなく，ただひたすら4×4パターンを描画します。その性質から，画面を一色で塗り潰す場合などに使用されるのでしょう。

最後は$3100です。これが一番複雑で，私が最後まで悩んでいたところです。$3000のようにフラグを使用しますが，今度は2ビットずつ判別していきます。さらに内部的(ファイル内ではなくプログラム内)に3つのフラグを持ち，それらの組み合わせでパターンを描画していきます。

とりあえず，それらの内部フラグがなにを意味しているかは置いておいて，実際の挙動を見てみましょう(図4)。一見でたらめに見えますが，よく見てみると，内部フラグの値にある規則を見出すことができます。まずf1とf2の関係ですが，両方とも同時に1になることはないとわかります。次に

リスト2

```
 1:
 2: unsigned short      pat2x2[256][4]; /* 2×2パターン格納領域 */
 3:
 4:    char      *src;    /* シネパックデータ格納領域 */
 5:    int       id;      /* 識別子 */
 6:    int       len;     /* ブロック長 */
 7:    unsigned int       flag;
 8:    int       no, i, j;
 9:    unsigned char      Y[4];
10:    char      Cb, Cr;
11:    int       R, G, B;
12:
13:    id = *(((unsigned short *)src)++);
14:    if( id==0x2100 ){
15:          len = *(((unsigned short *)src)++);
16:          len -= 4;                              /* 識別子とブロック長のぶん */
17:          for( no=0; len>0; ){
18:                flag = *(((unsigned int *)src)++);
19:                len -= 4;                        /* フラグのぶん */
20:                for( i=0; i<32; i++, no++, flag<<=1 ){
21:                      if( flag&0x80000000 ){
22:                            for( j=0; j<4; j++ ) Y[j] = *(src++);
23:                            Cb = *(src++);
24:                            Cr = *(src++);
25:                            len -= 6;        /* Y*4 とCb Crのぶん */
26:                            for( j=0; j<4; j++ ){
27:                               R = (Y[j]+1.40200*Cr)/8;
28:                               G = (Y[j]-0.34414*Cb-0.71414*Cr)/8;
29:                               B = (Y[j]+1.77200*Cb)/8;
30:                               if( R<0 ) R = 0; else if( R>31 ) R =
31;
31:                               if( G<0 ) G = 0; else if( G>31 ) G =
31;
32:                               if( B<0 ) B = 0; else if( B>31 ) B =
31;
33:                               pat2x2[no][j] = rgb( R, G, B );
34:                            }
35:                      }
36:                }
37:          }
38:    }
```

これは3DOのデータ

f2とf3はというと，こちらも両方が1になることはありません。それでいながら，f1，f2，f3のうち，少なくともどれかひとつは1になります。

さて，ここまでわかったところで，あるところで2×2パターンを描画したい場合を考えます。この場合，フラグで'11'を与えてやれば，内部フラグがどのような状態であっても，2×2パターンを描画できることがわかります。では今度は，4×4パターンを考えてみましょう。いま，仮にf2=0だったとすると，フラグで'10'を与えてやれば，4×4パターンを描画できますね。では，f2=1だったとすると？ 先ほどの内部フラグの条件から，f1もf3も0となりますから，今度はフラグで'01'とすればいいわけです。じゃあ，スキップしたい場合はどうでしょう。もしf1かf3が1ならば，'01'でいいですね。しかし，f1もf3も0（つまりf2=1）だったら？ どうやらそのときは方法がないようです。

そこでひとつ前に戻って考えてみます。もし直前が2×2パターンだったらどうでしょう。フラグは'11'で，内部フラグは不定なので，そのときもf2=1だったはずです。さて，その下の'10'の場合に目をやると，2×2パターン＋スキップという組み合わせがありますね。しかも都合がいいことに，f2=1の場合です。つまりf2=1で，2×2パターンの次にスキップがくる場合はこちらを使えばいいことになります。

同じようにして，スキップの直前が4×4パターンだった場合は，f2=1とならなければなりませんから，フラグが'01'でf1=0かつf3=0（f2=1）であったはずです。そこでフラグ'00'を与えてやれば，4×4パターン＋スキップという組み合わせになることはもうおわかりでしょう。最後に，直前もスキップだったら？ これはいいですね。一番下のスキップ×2を使えばいいのです。

これでおわかりでしょう。図のようにフラグを決めることで，2ビットで6通りのパターンの組み合わせを決めることができ，どんなパターンの並びにも対応できるのです（初期値はf1=1，f2=f3=0）。ただ，2ビットあればなにも考えなくても4通り定義することができるわけで，条件判断やフラグのセット/リセットを考えた場合，本当にこうしたほうが速いのかどうかは微妙なところですね。

最後に

サターンのシネパックでは，320×224ドット15コマ/秒というのが，標準的なようです。X680x0ではというと，残念ながらこれでは少し荷が重いようです。サターンは倍速CD-ROMとはいえインテリジェントですので，秒間300Kバイトという制限はあるものの，CPUから見ればオンメモリと変わりません。しかもメインCPUがRISC×2（ほとんどひとつしか使っていないという話もあるが）となれば，やむをえないといったところでしょうか。しかし，X68030のオンメモリでそれなりに動いているのを見ると，「SH2ってその程度なのか？」とも思えてきます。まあ，CPU全開で動いているのではないでしょうが。

ところで，3DOのPCMデータは圧縮されているようなのですが，どなたかフォーマットを調べた方いませんか？ もしいらっしゃいましたら，編集部までご一報ください。

図4 $3000のフラグの挙動

リスト3

```
 1:
 2: char *draw2x2( int, int, char *);/*4バイトから2×2パターン4個描画する関数*/
 3: char *draw4x4( int, int, char *);/*1バイトから4×4パターンを描画する関数*/
 4:
 5:     int     xsize, ysize;   /* 画像サイズ */
 6:
 7:     char    *src;           /* シネパックデータ格納領域 */
 8:     int     id;             /* 識別子 */
 9:     int     len;            /* ブロック長 */
10:     int     x, y, i;
11:     unsigned int    flag;
12:
13:     id = *(((unsigned short *)src)++);
14:     if( id==0x3000 ){
15:             len = *(((unsigned short *)src)++);
16:             len -= 4;                               /* 識別子とブロック長のぶん */
17:             for( y=i=0; y<ysize; y+=4 ){
18:                     for( x=0; x<xsize; x+=4, i--, flag<<=1 ){
19:                             if( i==0 ){
20:                                     flag = *(((unsigned int *)src)++);
21:                                     len -= 4;
22:                                     i = 32;
23:                             }
24:                             if( flag&0x80000000 ){
25:                                     src = draw2x2( x, y, src );
26:                                     len -= 4;
27:                             } else {
28:                                     src = draw4x4( x, y, src );
29:                                     len -= 1;
30:                             }
31:                             if( len<=0 ) break;
32:                     }
33:                     if( len<=0 ) break;
34:             }
35:     }
```

[特集]

Animation Now!

総論：映像環境への展望

SCSIによる究極の動画像環境

Nakano Shuichi 中野 修一

ここまで行ってきた各論をまとめてみる
アニメーション環境の新しい基礎作りを検討する
X68000による映像の可能性を探ってみよう

ということで，主に新しいSCSI2ボードを使って実現できるアニメーション環境についてそれぞれの立場でアプローチしてきたわけだが，ここではそれらを踏まえてSCSIを使った動画像環境の可能性をさらに追究してみたい。

具体的なことについては各論を参照してほしい。

まず，2つの立場から物事を考える必要がある。最終媒体をビデオにするか，ファイルにするかである。

プライベートなアニメーション制作環境のためであればハイエンドの機材を想定してかなり思いきったこともできる。しかし，要求されるスペックも非常にハイレベルなものになる。アニメーションファイルとしてやりとりするためのものであれば，ある程度の画質は確保しなければならないものの，より多くの環境で利用できるように実行負荷やファイルサイズを気にすべきである。

ハイエンド環境のアニメーション

本当のハイエンドだとマシンを2，3台必要とするのだが，さすがにそこまで要求するのは酷なので，ここで想定するマシンはX68030（クロックアップ済み）＋SCSI2ボード＋高速大容量HDDということにしておこう。SCSI2ボードが入手難だとはいうものの，いまとなってはそれほど無茶な環境でもないだろう。

使用するフォーマットは65536色無圧縮である。画質最重視にするとなまじの不可逆圧縮は採用できないこと，扱うのが可逆圧縮が効きそうにないデータだということがその理由である。

また，256色無圧縮という選択もある。絵柄によっては256色でも遜色ないこと，色数よりは解像度や再生レートを要求したいこともあることがその理由である。

基本的に，65536色無圧縮のシステムができあがれば，ほぼ自動的にその倍速で動作する256色のシステムができあがるので，256色システムの詳細についてはあまり気にする必要はないだろう。

無圧縮の世界

リアルタイムに圧縮されたデータを展開することは不可能ではないが厳しい。さらに画質を突き詰めると無圧縮になる。無圧縮ならどんなに描き込まれた画面でも普段と変わらない速度で再生ができるからである。

すでに，無圧縮のデータをSCSI装置から垂れ流すシステムにAMIがあるが，これはそのままでは今回のような用途には使用できなかった。転送速度が遅すぎたためである。

そこでSCSI2ボードの登場になる。AMIはデータをG-RAMに直接読み込んでいくタイプのツールなのだが，X68030のG-RAM周りはくせもので，実際にそれを行うとどうなるかというのはすでに解説されているとおりである。画面にときどきノイズが走る。X68030以外では一応問題なく使えるはずなので，そちらを採用することもできるのだが，X68000XVIでは転送速度がいまひとつ伸び悩んでいる。転送の理論限界値自体はX68000XVIのほうが速いはずなのだが，実測値ではX68030のほうが上回っている。数値としては，やや目標に足りないくらいのところである。

ベースはX68030にするしかない。しかし，今度はノイズが出る。AMIはシステム的にすでに完成されており，画質，速度とも必要な要件はすべて満たしていても，これではちょっと使えない。

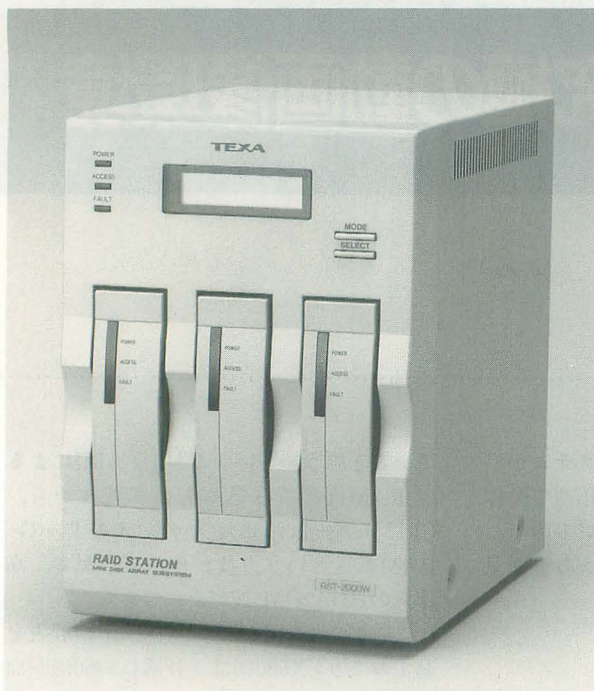
転送モードをソフトウェア転送にするとノイズは出ないのだが，必要な速度がまるで出ない。これについてもすでに解説されていると思う。ということで，AMIによるG-RAM直接読み込みという荒技から，CPUパワーに任せたメインメモリ→G-RAM転送という力技が浮かび上がる。高津氏のHDANIM.Xがそれである。

ここでちょっとフレームレートの解説をしておこう。X68000は1秒間に約60回画面を書き直している。画面書き換えはこの1/60秒単位で行わなければ画面がちらつくので，この時間を単位に表示システムは構成されることになる。毎回書き換えれば秒間60フレーム，1回おきなら30フレーム，以下，

2回おき	20フレーム/秒
3回おき	15フレーム/秒
4回おき	12フレーム/秒
5回おき	10フレーム/秒

のような具合になる。秒間10フレームというと凄く少なそうな印象を持つ人もいるかもしれないが，個人的には10フレームというのは滑らかに見える下限の数値だと思っている。10フレームを割ると実写での人の動きがもの凄く不自然になってくる。また，アニメは秒間24コマの3コマ撮りが基本なので秒間8フレームくらいでテレビアニメレベルの動きができると思っている人もたまにいるが，3コマ撮りのアニメでもパンニングやズーミングではちゃんと24フレームの動きをしているので，8フレームではとても足りない。アニメーションシステムでは最低15フレームくらいは確保したいところである。ちなみに，たとえ1秒間に120回画面に書き込んだところで60回分しか表示されないのでまったく無駄だ。

高津氏は15フレーム/秒というところまで確認しているわけだが，編集部でさらに追試験をしてみた。

マシンはX68030（35MHz）にMach-2の新しいBIOSを使い，ハードディスクはQuantumのEmpireよりひとクラス上のAtras(2Gバイト)を使用したシステムであ

RST-2000W

る。この状態でならなんとか20フレーム/秒が達成できた。

さらにハードディスクの最高峰としてディスクアレイというものも導入してみた。これはハードディスクを並列に置いたようなもので，カタログスペックからいくとドライブ単体で45Mビット/秒(5.6Mバイト/秒）の性能を持っている。これでさらにアクセスがドライブ3台に分散されて高速化されるはずなので，ボードの限界を知るには十分であろうと思われたのだが……。参考までに表1にX68030とSCSI2ボードでのDSKBENCH1.4の結果を挙げておく。アクセス速度は目を見張るほど速いのがわかる。しかし転送速度自体はかなりの数値を出してはいるものの，Quantumの2Gバイトハードディスクユニット Atrasを用いた場合にやや劣るという結果になってしまった。

このディスクアレイはSCSI3のWIDE SCSIに対応するなど速度的にも欲張った製品ではあるが，本質的には単に高速なだけのドライブではなく，高い信頼性を得るための製品である。1ドライブあたりの容量は1Gバイトなので，転送速度自体は2Gバイトのドライブには届かなかったのだろう(転送速度は主に記録密度と回転数に比例する)。インタフェイス的にも，性能を生かしきれなかったのかもしれない。

SCSI2ボードを使った場合，だいたい容量2Gバイトクラスのドライブなら，実測値でも4Mバイト/秒以上が確保できるようである。1Gバイトクラスでは QuantumのEmpireが広く出回っているが，このあたりだと少し微妙なところだ。

秒間20フレームなら一応HANIM.Xの標準的な再生速度と同じであるから十分実用レベルのシステムだといえるだろう。

31kHzではなんとか20fpsを出すことはできるようになったものの，タイミングが微妙なもので，15kHzモードでは18fps弱しか出ないという問題もあった。X68000の31kHzモードは秒間約55回画面を書き換える。ビデオ落としの際に使用する15kHzモードでは秒間約61回画面書き換えを行わなくてはならない。要するに画面を書くための時間が少しだけ厳しくなるのだ。最終的にビデオに落とすなら15kHzモードで20fpsを確保しなければならない。

この問題は，ビデオ出力時のオーバースキャン部分（画面外にはみ出してしまうところ）をカットして転送を軽くすることでなんとかクリアできることがわかった。

20fpsの次の目標は30fpsだが，これはなかなか厳しいところだ。究極のシステムという意味ではX68030 2台を使用し，同期させた256色ずつの画像を合成して65536色画像として出力するというシステムが最強であろうと考えられる。24ビットカラーでなくても，65536色できちんと誤差拡散などを行えばほとんど遜色のない画像を得ることは可能だ。無圧縮システムではどんな複雑な画像でも問題なく再生することができるのだから。1台あたりのデータ転送量も少なくて済む。このシステムなら無改造のマシンでも30fps再成が簡単に実現できるだろう。

メディアとしてのアニメ

ビデオ作品を作るだけがアニメーションの使い道ではない。しかし，アニメーションに関しては適当なファイルフォーマットを持っていないこともすでに述べたとおりである。あえていえば，SV.XやAMIがちゃんとあるのだが，残念ながらメジャーフォーマットにはなっていない。

データを持ち運べるようにするにはどうしても圧縮する必要がある。それはどうしても画像サイズや画質，フレームレートに影響を与えることになる。次に，データの問題だが，あまり複雑なデータを動かそうとしないことも重要であろう。まずはそこらへんから割りきらねばならない。

まず考えなければならないのは，再生の方式である。つまり，

オンメモリ

表1 RST-2000Wのベンチマーク結果

```
X680x0 DISK benchmark version 0.44 by bisco
Original program:
ASPI SCSI benchmark test V0.4
 copyright(c) by TsuruZoh Tachibanaya, Sep.02,1994

  Initiator is ID7     : SHARP   X68040           6411  Other-Port
  Target device is ID0 : TEXA    RST-2000         1.14  SCSI2
  512 Bytes per sector, capacity is 2012 MBytes.
------------------------+-------------+-------+-------+-------+-------+-------+
      Test mode         :    result   | Poor  |  OK   | Good  | Great |Superb|
------------------------+-------------+-------+-------+-------+-------+-------+
Test unit ready command |    3.8[ms]  :
No motion seek command  |    0.3[ms]  :*********************************
Average latency Time    |    5.5[ms]  :************************
------------------------+-------------+-------+-------+-------+-------+-------+
Sequential seek command |    0.7[ms]  :**********************************
Random seek command     |    0.8[ms]  :***********************************
------------------------+-------------+-------+-------+-------+-------+-------+
Seq.Read/Start   512B/rd|  388.6[KB/s]:***************************
Seq.Read/Start 16384B/rd| 3369.6[KB/s]:********************
Seq.Read/Start 65536B/rd| 4044.8[KB/s]:************************
------------------------+-------------+-------+-------+-------+-------+-------+
Seq.Read/ End    512B/rd|  378.6[KB/s]:**************************
Seq.Read/ End  16384B/rd| 2985.6[KB/s]:******************
Seq.Read/ End  65536B/rd| 3788.8[KB/s]:***********************
------------------------+-------------+-------+-------+-------+-------+-------+
Random  Read     512B/rd|   24.0[KB/s]:**********************
Random  Read   16384B/rd|  624.0[KB/s]:**********************
Random  Read   65536B/rd| 1728.0[KB/s]:*********************
------------------------+-------------+-------+-------+-------+-------+-------+
```

逐次再生
ブロック再生
の3つの形態である。

オンメモリならあまり考えることはないが，再生時間が短くなってしまう。誰もが12Mバイト実装しているわけでもない。

逐次読み込み逐次再生は応用性に富み，扱いやすい形式であるが，再生時の負荷はどうしても高くなる。

ある程度まとめてデータを読み，再生中に次のブロックを読むという方式は，データをうまく作ればもっともポテンシャルの高い方式だが，下手をすると再生速度のムラとして表れるためやや扱いづらい可能性がある。読み込み時の負荷にあわせて全体を整えるとパフォーマンスが発揮できないので難しいところではある。

ここではもっとも扱いやすい逐次再生方式に主眼を置いてみよう。たとえば，HANIMのデータを細切れにしてハードディスクに入れ，展開しながら足りなくなった部分を読み込んでいくというシステムは十分構築可能である。もちろん，DōGAレベルの画像を扱うことは無理だが。

このようなシステムでは時間管理を行うかどうかが問題となるだろう。音声をつけるなら時間管理が必須となる。要するに，速いマシンなら滑らかに遅いマシンでもカクカクしながらなんとか見れるということを保証するシステムである。しかし，時間管理するには，圧縮に前の画面との差分を利用することは禁じられてしまうので選択の幅は少し狭くなる。

時間管理せずに，ある程度以上速いマシンでは問題なく，遅いマシンではときどき重くなる……というシステムを選ぶことも理に反するわけではない。音声よりも画像を重視すればむしろこちらの選択になるであろう。

いずれにせよ，データ自体が流通するようになるにはファイルサイズを小さくする必要がある。ファイルサイズを小さくするためには画像サイズ128×128ドット程度で256色，といった基本フォーマットをちゃんと確立していくことが第一であろう。大きさとしては不満が残るところだが，こういうものは次第に慣れていくものだ。現在のDōGA CGAシステムなどで使われている256×256ドットにしてもX68000の表示能力からすればもの足りない気がしたものだが，その範囲でも十分な結果を出すことはできるのだ。

圧縮方式は無圧縮，ランレングス，ADPCM符号化，LZ符号化などが比較的軽そうな部類だ。いちばん軽いランレングスでもかなりつらいという問題もあるが。10MHz機を考えると無圧縮，ランレングス以外の選択はないだろう。となると，ランレングスの効きそうな絵を描くというのも大事になってくる。

無圧縮再生の場合，ファイル自体は強力な画像圧縮ツールでアーカイブしておき，再生時にベタのテンポラリファイルを作成して実行速度を上げるという手がある。もちろん，再生までにかなり時間がかかってしまうのだが。

ちょっと違う方面に目を向けてみよう。高津氏のHDANIM.XはもともとSCSI2ボードを使っても転送速度が足りないので，読み込んだデータを拡大処理しながら再生しようという目的で作成されたツールであった。結果としては圧縮データを展開しながら再生しようというのと似ている。

たとえば256×128ドットで作成された映像なら半分，128×128なら1/4のデータ量で済むので，ハイエンド環境でなくても再生は可能になる。解像度を落とすというのは，もっとも単純で効果的な不可逆圧縮ということもできなくはないのだ（画質の落ち方も凄いが）。

AMI Again

多少矛盾するが，AMIシステムはSCSIバスを使い無圧縮でデータを再生するものの，ちゃんとメディアとして成り立っている。128×128ドット65536色で秒間15コマの再生が可能な「メディア」である。256色なら軽く秒間30コマが出せる，または倍の時間の映像が収録できる。それはMOメディアごと流通させるというやや乱暴な方式をとったからである。

AMIのフォーマット自体は将来的に凄く速いSCSIが現れることを想定して作成されていたので，SCSI2ボードにはうってつけだったのだが，残念ながらそのままではハイエンドな，DōGAクラスの映像用ノンリニア編集環境としては適用できなかった。

標準データ速度は512Kバイト/秒というところが決定されており，MOを映像メディアとして使用するために必要なことはひととおり行われている。発表時は音声の出力はできなかったのだが，現在はとりあえず音声との同期も行えるようになっている。

時間管理はされていないが，最低位の機種と最低速度のデバイスでも動作が保証されているのでさして問題はない。HANIMの代わりに使うことはできなくても，HANIMにはできない長時間のデモ映像再生を実現することができる。

さらに，各種プログラムからデータを使用したり，AMI関係のツールを作成するのに必要なライブラリも用意されている。使い方次第ではアニメーションを盛り込んだゲームなどを作る際に使用できるシステムなのだが，肝心のデータ作成が実はいちばん難しい部分なのでいまだAMIを使ったゲームというのは現れていない（たぶん）。

Post AMI

AMI形式はデモ再生用などとしては優れたフォーマットである。しかし，ビデオ作成のためのフォーマットとしてはそのまま使用できない。そしてさらに最適化することができる。

ビデオへの収録を最終目的にする場合は，

ディスクアレイとは

今回は最高速ハードディスクシステムとしてディスクアレイを試用してみた。この装置はハードディスクユニットを複数個並列に設置したもので，全体でひとつのSCSI機器として扱われる。今回使用した日本テクサのRST-2000Wの場合，筐体に3個のドライブが設置されており，3つのモードで動作させることが可能である。

まず，RAID0は3Gバイトの容量を持ち，3つのユニットを並列にアクセスすることで大容量高速動作を実現するモードである。RAID3はドライブユニットのうち2つをデータ用に使い，残りの1個はパリティ用に使用するモードである。3つのドライブのどれかが使用不能になった場合でも壊れたユニットだけを交換すればデータは復活することができる。RAID5はさらに強い耐久力を持ったモードだ。

今回のように超高速ハードディスクとして使うのは本来の使い方ではない。

X68000シリーズでは役不足の感はあるが，パソコン用のハードディスクドライブというのはかなりチャチにできているので（あの値段ではしかたないとは思うが），何年も継続して使うのは危険である。安心に金をかけると人にはこういった製品が必要になるのだろう。少なくとも普通のハードディスクの何倍かは安全で，火事や大地震などではダメかもしれないが，普通に使っている限りは致命的なデータ損失が発生することはないだろう。

とにかく，ターミネータが接続されていないとエラーを発したりと，随所に安全対策が行き届いている。ちなみに，この製品では無停電電源装置の使用を推奨しているようだ。ごもっともで……。

画面サイズを上下左右20ドットずつくらいが画面外にはみ出てしまう。ここを最初から削ってしまうわけだ。G-RAMイメージそのままで扱いやすいAMIフォーマットを捨てることになるが，これでデータ量を1割以上減らせば，その分は確実にパフォーマンスになって表れてくる。

といったところで，X68000で実現できる最高の環境というものが見えてきた。X68030(33Dash以上)とSCSI2ボードを使い，ギガバイトクラスのHDDを媒体とすることで，256×256ドット65536色画像を秒間20フレームで連続表示することができるシステムである。画質は最高レベルをキープできるので，十分実用的なスペックであろう。

AMIと比べると目的とする環境は新しい次元のものになってくる。これはもはやノンリニア編集のための専用システムである。さしずめ，「SCSIによる究極の動画像環境（Ultimate SCSI Animation Graphic Interface)」といったところだろうか（略称は省略)。

1Gバイトあたり7分弱……。AMIがMO1枚に4～6分程度の画像を収録するシステムだったのだが，用途を考えれば大飯食いもしかたない。資源を無尽蔵に使うものの，ハードディスクにしても1Gから2Gバイトあたりが普及価格帯になりつつある現在ではそれほど無茶なシステムというわけではないだろう。

CGAとLVファイル

ここで，AMIと比べながら，方式自体は似たようなCGAファイルがなぜメディアになれなかったのかを見てみよう。

ざっと，

- 可搬性の有無
- 再生の保証
- 使用環境の違い
- プログラムからの再利用性
- ツールの充実度
- パフォーマンスの違い

といったものが浮かび上がる。

フォーマットすら公開されておらず，サポートツールもほとんどない。プログラムからの利用などはほぼ不可能に近い。SX-WINDOWからしか扱えないということも不利な要因である。

しかしなによりメーカー純正のものなのに映像用圧縮が考慮されていないという点が致命的であったのだと思う。それはなにも考えずに作ったというのとほとんど同義なのだから。

ビデオユニットの失敗

CGによるアニメーションの作成は誰にでも手軽にできるというものではない。となると，普通の人が扱うアニメーション画像としてもっとも手軽なのはビデオ映像の取り込みである。ビデオカメラからの入力であったり，テレビ放送の取り込みであったり，こういった画像をデータとして扱うことができればパソコンの用途も大きく広がっていくだろう。

X68000にはそのための機器としてカラーイメージユニット，ビデオイメージユニットが発売されている。カラーイメージユニットはかなり売れたはずだが，積極的に使用している人はたぶん少ないだろう。主に静止画の画像取り込み，あるいは無理やりの動画取り込みに使用されている。ビデオへの出力機構も持っているのだが，おそらくそのような用途で使う人はビデオボードのほうを使っているのではないかと思われる。

静止画ならともかく，動画像の取り込みというのはなかなか大変なことである。ビデオイメージユニットはその動画取り込みを謳い文句にして登場してきたわけだが，結局のところ，静止画を取り込む機能と高速なSCSIを組み合わせただけの製品であった。

すでに解説したように，ディスクからデータを読みつつ圧縮ファイルを展開表示することは困難だ。しかし圧縮せずに十分なデータ量を送るにはSCSIは遅すぎる。かといってオンメモリでは十分なことはできない。であるから，ハードウェアを付加してそれを解決しようと考えるのが自然な考え方というものだろう。動画像を扱うならこれがボトルネックになるということは火を見るより明らかなことであった。

高画質システムでは無圧縮データを使用しているように，ベタデータで保存すること自体はそれほど悪いことではないのだが，それは画像の大きさや再生フレーム数がちゃんと実用レベルに達していて初めて意味をなすものである。

ビデオイメージユニットでそれが実現できるのは外部にハードディスクを装着し，イニシエータとして動作させた場合のみ。取り込んだデータはいくらでも自由に加工できるとはいうものの，特に使い道は示されていない。ツールの使い勝手も悪く，X68000で活用する際に適当なサイズの画像を直接取り込むことができないので，あとから1枚ずつ加工する必要があった。これでは，そもそもなにを目的として作られたハードウェアなのかがまったくわからない。

「動画像をハンドリングできる」というの

HDアニメーション2種

お盆進行の過密スケジュールで進む9月号。ぎりぎりになってSCSIを使ったアニメーションシステムの投稿があった。福岡県の奈良原伸哉さんによるもので，ANMPLAY.XとAM2LOAD.Xの2種類のシステムである。

ANMPLAY.Xは差分，ランレングスなどを使って圧縮したデータをある程度まとめて読み込んで再生中に割り込みで次のブロックを読んでいくという感じのアニメーションシステムだ。条件にもよるのだろうがX68000XVI(24MHz)で秒間20フレームというHANIM.X並みの速度を実現しているのは立派である。

森山氏作の「A DRAGONFLY」からオープニング映像（本編はさすがに無茶だろう）をコンバートしてみた。黒地の画面にトンボが飛んでいくところである。

垂直同期を見ていないためか，あるいは2画面切り換えをしていないためか（直前の画面との差分を取っている可能性あり)，画面書き換えが見えてしまう。さらにテクスチャポリゴンが多くなるとかなりガクガクしてくる。4096色への減色を行うオプションを指定すると（ランレングスが効きやすくなる)，それなりに滑らかにはなる。絵の階調は少し落ちるのだが，4096色くらいあればそれほど気になることはない。

ノンテクスチャの映像（X68000芸術祭オープニングとか）ならかなり滑らかに動いていたのだが，このへんがランレングスの限界だろう。

テクスチャは使用しない映像用と割りきって使う分には非常によくできたシステムであるといえる。

もうひとつのAM2LOAD.Xは170×253ドット，256色無圧縮の画像を垂れ流していく形式のシステムで，秒間12フレームの画像が再生できる。ちょっと変な大きさだが，CRTCをいじって専用の画面モードを作っているので，見た目にはほとんど違和感はない。まあ，DōGAの標準的な画像（65536色を256色に落としたもの）を見慣れているのだからかえって画質は高く見えるくらいだ。さらにPCMとの同時再生もサポートされている。

実はAMIの画面モード拡張案に似たようなものがあったのだが，AM2LOADはちゃんと時間管理している。AMIと同じようにG-RAMに直接読んでいるのでほぼ同じパフォーマンスが得られる。MOからでも秒間12フレームは堅いはずだ。それでもしっかり時間管理をしている。

Oh!Xでの公開はされていないが，AMIはすでにAD PCMとの同時再生を行っているので，AM2LOAD.Xの機能的な部分はほぼAMIに吸収できることになる（どの機種でも再生できれば時間管理は不要になる)。ということでAMIのバージョンアップも行いたいところだが，作者多忙につき現在は見送られている。

アニメーション制作技術だけは進んでいるが……

が売り文句だが，だからといって「Quicktime」などの単語にお目にかかるのはMacintosh用のマニュアル部分でだけのことであった。X68000はCGAファイルを使えということなのだが，これがメディアとして成立するものではないこともすでに解説したとおりである。

映像をメディアとしてとらえた場合，画質は最重視されるものではない。CPUパワーが足りないことが目に見えているX68000の周辺機器としては，多少画質が劣ってもかまわないからハードウェア圧縮機構を備えるべきであっただろう（というより，値段を聞いたときにまさか圧縮機構がないなどとは夢にも思わなかった）。

汎用に作られているので，それを使ってできること自体はX68000専用のカラーイメージユニットよりも少なく，1枚絵の画質は上がったものの，CPUへの負荷は大きくかかり，フレームレートは格段に落ちている。「画質が格段に上がった」という1点を除けば，大幅にパワーダウンした製品といわれてもしかたがない。

互換性がなくて倍以上の値段のするものを作るときに，圧倒的な性能がなくては受け入れられるはずはないだろう。

問題なのはSCSIバスを映像用バスとして使用したということなのだが，こうなってくるとSCSIバスのトラフィックも結構問題になってくるだろう。アニメーション再生しながらマルチタスクで別の作業などをするといったことが困難になる。アニメーション再生のたびに1基しかないSCSIポートが始終占有されていてはシステムのパフォーマンスは著しく低下する。SX-WINDOWが仮想記憶をやってなくてよかったというところだろうか。

ところで，無圧縮で最高の状態の画像を保存/再生するにはどうすればいいのだろうか。512×512ドットで65536色の画像を秒30フレーム保存するには秒間15Mバイト程度の速度の転送が必要になる。これはFAST SCSIの限界を超えた数値だ。ハードディスク自体は中村氏の記事にあるように8Mバイト/秒くらいまではついてこれるらしいのだが。

となれば，あとはSCSI2ボードを2枚並列で動作させて2台のハードディスクを同時に使う……という手が考えられる。残念ながら，X68000のバスの速度（10MHzの16ビット拡張スロット）では現状の速度でもかなり限界に近いようだ。ハードディスクの性能の半分しか引き出せない。せめてX68030で32ビット専用スロットがあったら……というのが返すがえすも残念なところだ。

ハードウェア圧縮

無圧縮での保存が不可能とはいわないまでも困難であれば残るはハードウェア圧縮しかない。このへんの話はX68000シリーズとはもはや無関係だが，最近の世の中の動きというところを見ておいてもらいたい。

PC/AT互換機にはすでにハードウェア圧縮機能を持ったビデオキャプチャカードがいくつも市販されている。多くは独自フォーマットでテンポラリファイルを作ったりする形式のものだが，最近はいきなりMotionJPEGあるいはMPEGで保存するものも現れてきている。

昨今のゲーム機ではMotionJPEGデコーダを内蔵したものも増えてきた。PlayStation，PC-FX，3DO M2アクセラレータあたりで採用されているのでもうお馴染みかもしれない。

MotionJPEGは基本的には画面間の差分を取らない方式だが，MPEGの2倍の情報量を使うことでかなりの画質を確保できる。差分を取らないので逆再生や早送りといったトリックプレイにも対応できる。データ作成や編集にも都合がいいのでパソコンではMPEGよりも普及する可能性が高い。

MPEGはビデオCDとして登場しているのでこちらもお馴染みかもしれない。CD-ROM1枚で74分という長時間再生が可能だ。

前の画面からの動き予測に対する符号化を行っているので圧縮効率が高いがトリックプレイなどは困難である。MPEG1の最大の欠点はデータ転送レートが固定されていることだろう。それも150Kバイト/秒というCD-ROM速度なので，画質を確保することが難しい。

MPEG2なら情報量としては問題ない。ただし，MPEG2というのは天井が高いのでMPEG2デコーダという名をつけるためには非常にたくさんの処理をサポートしなければならないことが予想される。ハイビジョン対応はともかく，基本機能だけでも普及価格まではかなり遠そうだ。

とまあ，MotionJPEGのほうが使い勝手がよいのだが，データ自体のフォーマットとしてはMPEGに落ち着くのではないかと思う。MotionJPEGで作って，最終的にMPEGに落とすという形態が理想的である。データ量の問題と再生環境がより多くなるということは重要なことだ。

さて，MPEGにしてもある程度までならソフトウェアでデコードできることは確かだ。ハイエンドのAT互換機ではほぼ完全なMPEGのソフトウェア再生ができる（という）。JPEGならさらに軽いだろうから（おそらく），CPUが十分に速ければMotionJPEGのソフトウェアデコーダを作成することは可能かもしれない。X68030の場合なら，ハイエンド仕様に改造してDSPボードを付加してもまだ苦しいところだろうか？

CPUが速くなればソフトウェアでデコードすることも可能にはなることはわかる。しかし，リアルタイムにエンコードするとなるとちょっと無理だろう。ここにきて，従来はかなり高価なものだったエンコーダチップも大量生産による低価格化が進んできている。やや古いがMotionJPEGのエンコーダボードが5万円とかPC-9821でもMPEGエンコーダが（高価だが）で利用できるようになってきているなど，すでにハードウェアでアニメーション録再をサポートする傾向になってきている。もう2年もすればビデオ代わりにハードディスクにテレビ録画するようなことも当たり前のように行われるようになるのだろう（ランニングコストはともかく）。

このようにパソコンのマルチメディア化は急ピッチで進みつつある。メインストリームから取り残されていてはビジネスマシンに笑われるような時代だってくるだろう。

MotionJPEG，MPEG，こういったもので画像がメディア化される時代は目前である。その先にある世界はまだ見えていないが，あくまでも個人的な見解としては，フラクタル圧縮が残りそうだとは思う。MPEG4が先か，あるいはMPEG4自体がフラクタル圧縮になるのかはまだわからないのだが。

WE WANT YOU!

Oh!Xは，読者の皆さん1人ひとりの力が作り上げていく雑誌です。あなたも誌面作りに参加してみませんか？

投稿大募集

Oh!Xでは読者の皆さんによる投稿作品を常時募集しています。

未発表の作品であれば，グラフィック，音楽，システムプログラム，ツール，ゲーム，ハードウェアなどジャンルを問いません。機種についても特に限定はしませんが，雑誌の性格上扱いにくい場合もあります。

誌面に載りきらない大きなアプリケーションなどはディスクメディアを使って配布することが考えられます。その形態のひとつはご存じ付録ディスク，そしてもうひとつは別冊形式によるものです(発売中の「Z-MUSICシステムver.2.0」に続き，今後もいくつかのOh!X BOOKSシリーズが予定されています)。

また，「こんなものを作ってみました」といったものでもかまいません。気軽に作品を送ってみませんか。

投稿募集要項

1) お送りいただくプログラムには，住所，氏名，年齢，職業，連絡先電話番号，機種名，使用言語，動作に必要な周辺機器，パソコン歴などを明記のうえ，封書の宛先の最後には「Oh!X LIVE」「全機種共通システム」「投稿ゲームプログラム」など，プログラムの内容を明確にご記入ください。

2) 投稿されるプログラムには詳しい内容を記入した原稿を同封してください。ディスクの中にドキュメントファイルの形式でのみ記述している方がいますが，郵送時の事故などでメディアが破壊されることもありますので，必ず文書を添えるようにしてください。変数表，メモリマップ，参考文献などの情報があればなお結構です。また，掲載に際しては，プログラムやデータ原稿に対して加筆修正をさせていただくことがあります。

3) お送りいただくプログラムは事故防止のため最低2回はセーブしておいてください。基本的に原稿などの返送はいたしませんので，あらかじめご了承ください。

4) ハード製作関係の投稿については，最初は内容のわかる原稿のみお送りいただければ結構です。その後，当方で製作物が必要だと判断した場合には改めてご連絡いたします。

5) 作品の採用については，掲載号が決定した時点で当方より連絡いたします。特にツールやハード関係などの作品は特集内容などを考慮したうえで採用決定されますので，結果を連絡するまで時間がかかる場合があります。

6) 投稿いただいたプログラムにバグなどが発見された場合は，新しいプログラムの入ったメディアと一緒に文書にてご連絡ください。

7) 掲載されたプログラムに対しては当社規定の原稿料をお支払いいたします。また，投稿されたプログラムの著作権などはすべて制作者に保留されますが，いわゆる「フリーソフトなどとしてネットにアップする」ことなどを希望される場合には，必ず事前に編集部までご連絡ください。なお，一般的モラルとして，他誌との二重投稿，または他誌に掲載されたプログラムの移植などは固くお断りいたします。

その他，不明な点は編集部までお問い合わせください。

Oh!X編集部 ☎03(5642)8122

協力スタッフ募集

Oh!Xでは誌面作りに参加していただく協力スタッフを募集しています。

スタッフとして活動する熱意があり，東京近郊にお住まいの方でソフトバンクに来社可能な方。時間的束縛は特にありませんが，ある程度時間に余裕がある方に限ります。基本的に学生を対象にしていますが，時間的余裕と余力が十分にあれば社会人も可とします。ただし，18歳未満の学生および浪人生の方については採用予定はありません。

応募要項ですが，ライター希望の方はOh!X誌面1ページ分相当(2500字程度)の自由論文に自己紹介文を添えて「Oh!Xスタッフ希望」係までお送りください。

また，文章力には自信がないけどプログラムなら……という方でも技術スタッフとして参加していただく場合があります。こちらを希望の方は，自由論文の代わりにこれまでに制作した自作プログラムとその解説などを一緒に応募してください。

書類選考後，採用の方にはこちらからご連絡いたします。

すべての読者へのお願い

いまはまだ何もできないけれど，いつかは……と思っているアナタにも，いますぐできるいちばん重要なことがあります。アンケートハガキへのご協力です。Oh!Xの誌面の方向性は，このアンケートで寄せられた読者のご意見をもとに決定されています。

皆さんからの熱いメッセージをお待ちしています。

そして，宛先

〒103 東京都中央区日本橋浜町3-42-3
ソフトバンク株式会社
Oh!X編集部 ○○○○係

イラスト投稿の規定

サイズはハガキ大(A6判)からB5判くらいまでを目安としますが，取り扱いの手間や現実的な問題としてハガキ大を一応の標準とします。いずれにせよ，掲載時にはかなり縮小されることを考慮して描いてください。

一応の推奨形式は以下のとおりです。

1) ハガキ大のケント紙で郵送

ハガキでも結構ですが，たまに裏面にも消し印が押される危険があります。

2) 黒一色(薄ズミ不可)

墨汁は汚れの原因になることがあります。製図用インクがおすすめです。原稿は縮小されますのでスクリーントーンの80, 90番台(レトラセットの場合)や色の濃すぎるものなどについての再現は保証しかねます。また，残念ながら，カラー原稿はごくたまにしか掲載されません。

内容に関して特に規制はありませんが，季節ものについては，掲載が予想される時期を考慮して早めに送ったほうが有利になることがあります(年賀状は例外)。

皆さんの力作をお待ちしております。

BACK ISSUES バックナンバー案内

ここには1994年9月号から1995年8月号までをご紹介しました。現在1994年4〜12月号，1995年4〜8月号の在庫がございます。バックナンバーはお近くの書店にご注文ください。定期購読の申し込み方法は128ページを参照してください。

1994

9月号
特集　SX-WINDOW環境セットアップ

連載　響子 in CGわ〜るど/ショートプロ/ハードコア3D
ローテク工作/DōGA CGアニメーション講座/善バビ
システムX探偵事務所/ファイル共有の実験と実践

- 新製品紹介　X68030 D'ash/MJ-700V2C
- 新刊紹介　X680x0 TeX

LIVE in '94　LOVE IS ALL/HELL HOUND/踏切の通過音
THE SOFTOUCH　餓狼伝説SPECIAL
全機種共通システム　怪しいZ80の使い方(テクニック編)

10月号
特別企画　もみじ狩りPRO-68K

連載　響子in CGわ〜るど/ショートプロ/ハードコア3D
TeX入門講座/ゲーム作りのKNOW HOW/善バビ
猫とコンピュータ/ファイル共有の実験と実践

- 特別付録　もみじ狩りPRO-68K(5″2HD)
- 新製品紹介　F-Card V5 for x68k

LIVE in '94　イース2/MSX用GRADIUS2/NATURE
THE SOFTOUCH　スーパーストII/スターラスター 他
全機種共通システム　怪しいZ80の使い方/ゲーム作成講座(3)

11月号
特集　STEP UP BASIC

連載　響子 in CGわ〜るど/ショートプロ/ハードコア3D
TeX入門講座/DōGA CGアニメーション講座
システムX探偵事務所/ローテク工作/善バビ

- 新製品紹介　BJC-400J/X680x0 Develop. & libcII
Free Software Selection Vol.2

LIVE in '94　ダーク・スペース/ENDLESS RAIN/レナのテーマ
THE SOFTOUCH　スーパーストII/餓狼伝説SPECIAL
全機種共通システム　B-GALETS2

12月号
特別企画　XL/Imageお試し版＋α

連載　響子in CGわ〜るど/ショートプロ/ハードコア3D
ファイル共有の実験と実践/DōGA CGアニメーション講座
システムX探偵事務所/ローテク工作/TeX入門講座

- 特別付録　XL/Imageお試し版＋α(5″2HD)
- 新製品紹介　H.A.R.P/XDTP SX-68K

LIVE in '94　幻想即興曲/きまぐれ　オレンジ☆ロード 他
THE SOFTOUCH　魔法大作戦/スーパーストII
全機種共通システム　シューティングゲーム作成講座(4)

1995

1月号(品切れ)
特集　割り切って使うCD-ROM

連載　響子 in CGわ〜るど/ショートプロ/ハードコア3D
ファイル共有の実験と実践/DōGA CGアニメーション講座
システムX探偵事務所/ローテク工作/TeX入門講座

- CD-ROMドライブ紹介　CS-CD301X/CDS-E/SCD-200
- 新製品紹介　X68000XVI用アクセラレータXellent30

LIVE in '95　ぷよぷよ/ジムノペディNO.1/PRIME
THE SOFTOUCH　バックランド/上海 万里の長城/魔法大作戦
餓狼伝説SP 特別編/スーパーストII 特別編

2月号(品切れ)
特集　MicroProcessingUnit

連載　響子 in CGわ〜るど/ショートプロ/ハードコア3D
SX-BASIC公開デバッグ/DōGA CGアニメーション講座
システムX探偵事務所/SX-WINDOWによるDTP

- 特別企画　最新ゲーム機を見る
- 新製品紹介　Datacalc SX-68K/シャーペンワープロパック
- 1994年度GAME OF THE YEARノミネート作品発表

LIVE in '95　サムライスピリッツ/AFTER SCHOOL/白鳥の湖
THE SOFTOUCH　スーパーストII 特別編

3月号(品切れ)
特集　SoundEffects

連載　響子 in CGわ〜るど/ショートプロ/ハードコア3D
システムX探偵事務所/ファイル共有の実験と実践
ピコピコエンジン活用講座/SX-WINDOWによるDTP

- SX-WINDOW用ユーティリティ　どっち.X

LIVE in '95　魔法のプリンセスミンキーモモ/別れの曲
ファイナルファンタジーII/宇宙戦艦ヤマト完結編
THE SOFTOUCH　ディグダグ/ディグダグII/VIEW POINT
全機種共通システム　S-OSシステムコールライブラリ

4月号
特集　Let's Play Wonderful GAME

連載　響子 in CGわ〜るど/ショートプロ/ハードコア3D
システムX探偵事務所/ファイル共有の実験と実践
DōGA CGアニメーション講座/ローテク工作

- 1994年度GAME OF THE YEAR発表
- 新製品紹介　TS-6BSImkII/MJ-5000C/MATIER ver.2.1

LIVE in '95　天聖龍/ファイナルファンタジーVI/
ANOTHER DAY/ハートオブザマッドネス
全機種共通システム　S-OSねちねち入門(1)

5月号
特集　Realize Graphic

連載　響子 in CGわ〜るど/ショートプロぱーてぃ
ローテク工作実験室/SX-BASIC公開デバッグ
システムX探偵事務所/ANOTHER CG WORLD

- 特別付録　Oh!電脳倶楽部
- 新製品紹介　フォント&ロゴデザインツール

LIVE in '95　ドラゴンセイバー/ミッドナイトレジスタンス 他
THE SOFTOUCH　ボンバーマン ぱにっくボンバー
全機種共通システム　S-OSねちねち入門(2)

6月号
特集　Open the SX-WINDOW

連載　響子 in CGわ〜るど/ハードコア3Dエクスタシー
DōGA CGアニメーション講座/ローテク工作実験室
システムX探偵事務所/ショートプロぱーてぃ

- 特別企画　X68000周辺機器パワーアップ計画
- 新製品紹介　Xellent30s/学研統合電子辞書 for SX-Window
- 第6回アンケート分析大会

LIVE in '95　クリティカルポイント/THE SUMMER OF '68 他
全機種共通システム　S-OSねちねち入門(3)/BLOCK DOWN

7月号
特集　Optimizing Method

連載　響子 in CGわ〜るど/ハードコア3D/ファイル共有
DōGA CGアニメーション講座/ショートプロぱーてぃ
システムX探偵事務所/ANOTHER CG WORLD

- THE USER'S WORKS SPECIAL
- 新製品紹介　PDドライブLF-1000

THE SOFTOUCH　バラデューク
LIVE in '95　クロノ・トリガー/SUPER MARIO BGM集 他
全機種共通システム　FE ver.1.0

8月号
特別企画　暑中見舞いPRO-68K

連載　響子 in CGわ〜るど/(善)のゲームミュージック
DōGA CGアニメーション講座/ショートプロぱーてぃ
システムX探偵事務所/ANOTHER CG WORLD

- 特別付録　暑中見舞いPRO-68K(5″2HD)
- 新製品紹介　SCSI2ボードMach-2/DSPボードAWESOME-X
CD-ROMドライブCDG-TX4

LIVE in '95　淡紅色の夢/Tomorrow never knows 他
全機種共通システム　IF ONLY

SIDE A

処理系を整理してみる

Tan Akihiko 丹 明彦

プログラムを機能ごとに分け，システム全体を整理する
大きなプログラムを効率よく作成するためには，必ず必要になることだ
今回は，ゲームシステムの全貌と座標系の見直しを行っていく

とりあえず車の直線運動は片づき，さらに複雑な運動の記述に進みたいところだが，その前に一度システムを整理しておきたい。整理するといっても，誌上ではソースコードを公開していないので具体的な実装に踏み込むことは避け，処理系をどういう構造にするかということを解説する。

システム概観

3次元ドライブシミュレータという処理系を構成する要素は，大まかにいって次のようになると考えられる。

・コースデータ処理

モデリングされたコースデータを読み込んで表示しやすい形式に整理する処理。車両挙動シミュレーションの場合，路面との接地状況を調べるための情報を返す処理も含まれる。過去に解説したが，表示用の路面と接地判定用の路面は別にもっておくと検索が高速に行えるので好都合である。

・ユーザー入力処理

いうまでもないが，キーボードやジョイスティック，マウスなどから入力データを取る処理。デバイスの違いを吸収する形で書くのが望ましい。

・車両挙動処理

ユーザーからの入力を受けとり，ある瞬間の車の状態から次の瞬間の状態を求める処理。路面との接地状況を調べるため，コースデータにもアクセスする。

・タイム計測処理

車のある瞬間の位置と次の瞬間の位置から，その車がタイム計測ラインを通過したかどうかを判定し，通過したならその時刻を正確に求める処理。コースデータに用意されているタイム計測ラインの情報を利用する。

・ポリゴン表示処理

車両挙動によって決定された視点と視線方向から視野を求め，視野に入っているコースや車などのポリゴンを表示する処理。極端に遠いポリゴンは表示を省略して処理速度を稼ぐのが常套手段。

・サウンド処理

エンジンの音やロードノイズ，タイヤのスキール音などを発生させる処理。プレイヤーの車以外にも音を発生する物体がある場合は，視点からの方角や距離，速度によって変化した効果音を鳴らす。

そのほか，レースということにするならそのルールに沿ったマネージメントを行う処理や，計測したタイムやタコメーターなどを表示する処理，さらに複数台の車が走るなら，それらの間の衝突判定を行う処理も必要になる。ハードウェア的に贅沢な環境なら，ステアリング反力などのフィードバック処理が入ってくるだろう。

大きなシステムを書く際の心得

上に挙げた諸々の処理はそれぞれ独立させて書くべきである。いわゆるモジュール化というやつである。もちろん，それらの処理の間で情報のやりとりが発生するのだが，インタフェイスをきちんと決め，互いの処理が混ざらないようにしなくてはならない。

こうすることは分業する際には絶対に必要だし，たとえ分業していなくてもバグの発生を抑えやすくなる。ひとりで作っていると，ついつい安易な場所にさまざまな処理を押し込めてしまうものだが，あとで必ず泣きをみるのである。

座標系を考え直す

個々の処理を独立させる話と関連して，ローカルルールを排除する話もしておきたい。本連載では，X68000/030とそのポリゴンシステムSLASHをベースとしてプログラムを作っている。ゆえに，ある程度SLASHのルールを意識したプログラムを書くこ

とは必要だが，1から10までSLASHに合わせる必要はないし，そうしないほうがいい場合もある。

なんの話かというと，座標系である。SLASHはX68000のグラフィック座標，すなわち画面の座標系に沿ってX軸とY軸を決めている。X軸正方向が右，Y軸正方向が下，そしてZ軸正方向は前と規定されている。コースや車のモデリングも車両挙動シミュレーションもこの座標系に沿って行われてきた。

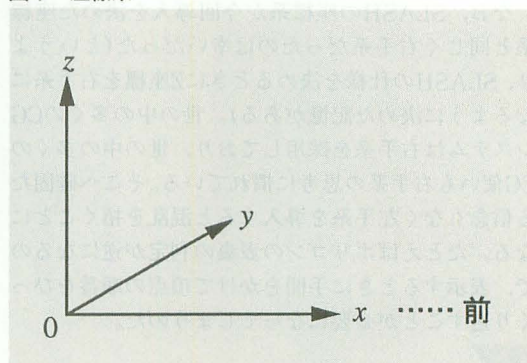
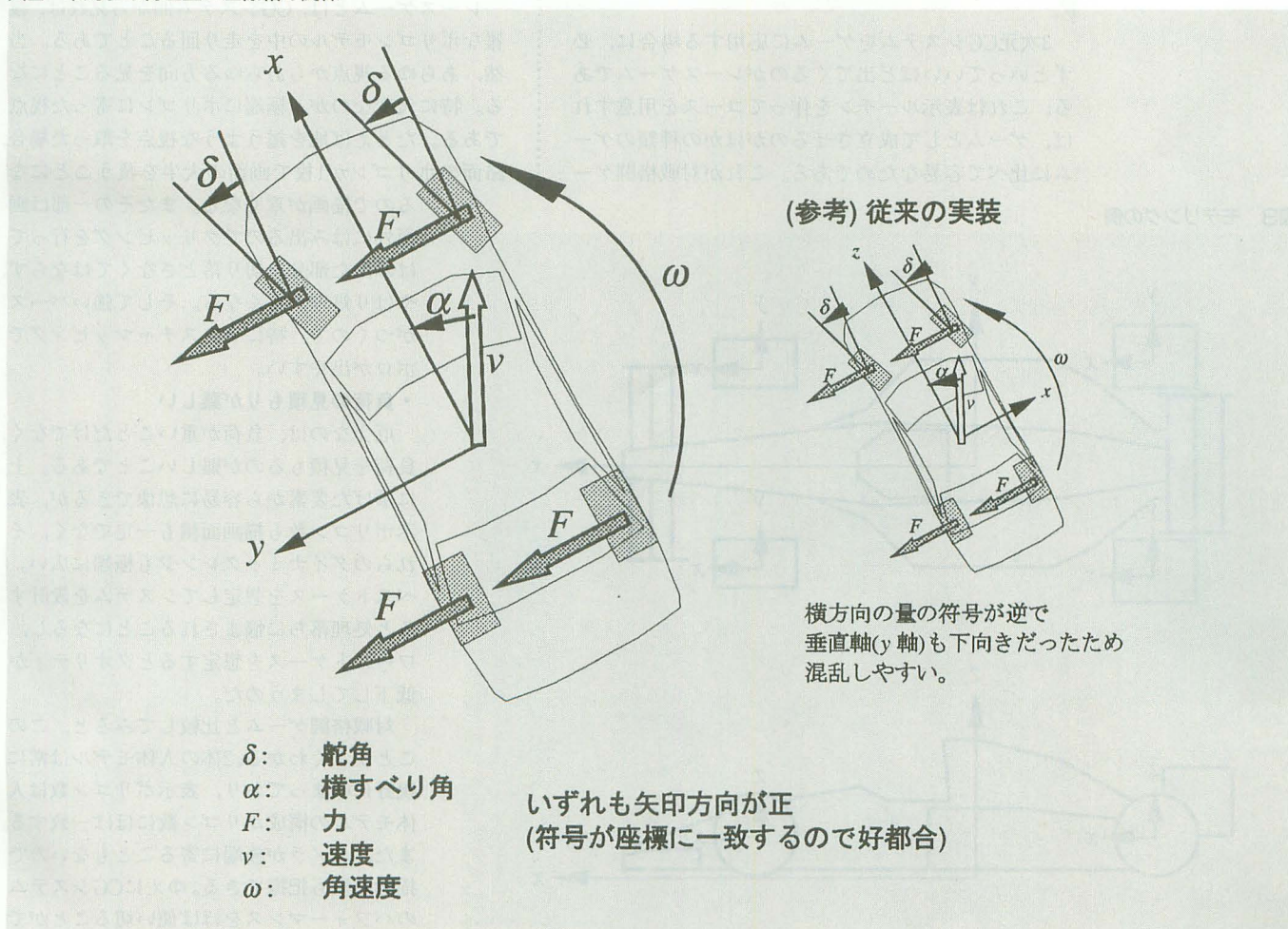
ところがこの座標系は個人的には扱いづらいと感じる。あくまでも個人の感覚の話でしかないのだが，下向きが正というのになかなか慣れることができないのだ。特に車両挙動シミュレーションは大量のあらゆる向きのベクトルを扱うため，正負の感覚がしっくりこないとミスのもとにもなるのだ。

そこで発想の転換が必要になる。ポリゴン表示システムのルールをシミュレーションアルゴリズムにまでもち込む必要はないのだ。シミュレーションはもっと感覚に馴染む座標系を使って，その結果だけをSLASHの座標系に合わせて使えばよいのである。

今後は図1に示したような座標系を使うことに決めた。前がX座標，左がY座標，上がZ座標である。前がX座標というのは若干わかりにくい気もするが，左方向に座標軸を取ることのメリットからこうした。図2を見ていただきたい。旋回を起こすのは横向きの力なのだが，なかでも正の角速度を伴った旋回を起こすのは左向きの力なのである。符号はなるべく揃っているほうがなにかと便利と考えられる。これもいくつか車両挙動シミュレーションを書いてきた経

図1　座標系

図2　車周りの物理量と座標軸の関係

いずれも矢印方向が正
(符号が座標に一致するので好都合)

験のなせるわざというものであろう。

ついでに，モデリングの座標系についても図3に示すように合わせることにした。SLASHのポリゴンリストを作る段階で座標を入れ替えてもいいし，表示する段階で座標や回転角を入れ替えるようにしてもいい。モデリングデータをこの座標系に従って用意するということである。以前はY座標が下向きだったので，方眼紙を見ながらモデリングするのが大変だったのだ。

なお，SLASHの座標系が今回導入を決めた座標系と同じく右手系だったのは幸いだった(というより，SLASHの仕様を決めるときにZ座標を右手系になるように決めた記憶がある)。世の中の多くのCGシステムは右手系を採用しており，世の中の多くのCG使いも右手系の思考に慣れている。そこへ確固たる信念もなく左手系を導入すると混乱を招くことになる。たとえばポリゴンの表裏の判定が逆になるので，表示するときに手間をかけて頂点の順番をひっくり返すことが必要になってしまうのだ。

レースゲームの意外な難しさ

3次元CGシステムをゲームに応用する場合に，必ずといっていいほど出てくるのがレースゲームである。これは表示ルーチンを作ってコースを用意すれば，ゲームとして成立させるのがほかの種類のゲームに比べて容易なためである。これが対戦格闘ゲームだと，ステージと人体モデルを用意した時点からゲームとして成立するまでの距離がものすごく大きいのである。

ところが，レースゲームというものはCGシステムにありがちな応用例であるにもかかわらず，CGシステムに対してもっとも大きな負荷を強いる用途のひとつなのである。理由はいくつかある。そして，それらの負荷を緩和するために多くのテクニックやノウハウが存在し，その影響はシステムの根本的な設計にまで及ぶ。経験がものをいう世界でもある。

・表示するポリゴンの選択処理が重い

レースゲームのコースは広い。コースを構成するポリゴンは数万ポリゴンに及ぶが，そのうち同時に表示されるのは数百から数千ポリゴン。CGシステムのパフォーマンスには限界があるので，一定のフレームレートをキープしようとするなら表示ポリゴン数を減らすしかない。前述したように表示されるのは視野に入っており視点から十分に近いポリゴンだけである。これを選択するのにそれなりの処理量が必要となる。

・ポリゴンの表示形態が極端になりやすい

レースゲームとは，CGシステム側から見れば，複雑なポリゴンモデルの中を走り回ることである。当然，あらゆる視点からあらゆる方向を見ることになる。特に大変なのが，極端にポリゴンに寄った視点である。たとえば地を這うような視点を取った場合，路面のポリゴンが1枚で画面の大半を覆うことになるので描画が重くなる。またその一部は画面外にはみ出るのでクリッピングを行ってはみ出た部分を切り落とさなくてはならず，やはり処理が重くなる。そして強いパースがつくので，特にテクスチャマッピングでボロが出やすい。

・負荷の見積もりが難しい

厄介なのは，負荷が重いことだけでなく，負荷を見積もるのが難しいことである。上に挙げた要素から容易に想像できるが，表示ポリゴン数も描画面積も一定でなく，それらのダイナミックレンジも極端に広い。ベストケースを想定してシステムを設計すると処理落ちに悩まされることになるし，ワーストケースを想定するとクオリティが低下してしまうのだ。

対戦格闘ゲームと比較してみると，このことがよくわかる。2体の人体モデルは常に視野に収まっており，表示ポリゴン数は人体モデルの構成ポリゴン数にほぼ一致する。また，カメラが極端に寄ることもないので描画面積も把握できる。ゆえにCGシステムのパフォーマンスをほぼ使い切ることがで

図3　モデリングの例

きるのだ。だからといって格闘ゲームを作るのが楽ということはなく，ちゃんと別の部分で苦労しなくてはならないことになっている。世の中とはよくできているものなのだ。

効率のよいコース表示を行うために

以上のことから，コース表示はデータ構造を含めて相当注意深く設計しなくてはならない問題であるということがわかる。単純にCGモデリングツールから出てきたデータを表示ルーチンに通すのではなく，データの性質を把握して不要な処理を省略してチューニングを行わなくてはならない。

データの性質を把握するというのは，たとえば「レースゲームは車が地面の上を走り，それを地面の上にある視点から見るものである。したがって，登場する物体は必ず地面より高い優先順位で描画される」ということである。

もちろん，コースにアップダウンやバンクや立体交差がある場合は，この仮定も通用しない。しかしアップダウンが緩い場合ならおおむね大丈夫だし，バンクしているコースの外側に視点がこないならやはり問題は起きない。

つまり，どういうレースゲームにするかというレベルでの決定がシステム設計に影響を及ぼす。「ハードドライビン」にするか「インディ500」にするかで，コースも表示ルーチンも変えるべきなのだ。

図4は，「なだらかなサーキットにまばらな地上物と車が載っている」ようなレースゲームを想定した場合のオブジェクト種別の例である。

地表および路面はおおむね水平面に収まっており，ほかのどのオブジェクトよりも下にある。また，ある部分が別の部分を覆い隠すということもないと考えてよい。よって，地表と路面を構成するポリゴンは表示の際に最初に描き，Zソートも一切行わない(一般にZバッファをもたないシステムの場合，ポリゴンは視点からの距離でソートし，遠い順番に描画する)。

壁や橋など，地面に固定されていて，かつほかのオブジェクトと重なって表示される可能性の高いオブジェクトは，表示する際にZソートが必要になる。これを怠ると，壁の向うにいるはずの車が透けて見えたりすることになる。個々のオブジェクトが十分に離れていると仮定するなら，ソートはポリゴン単位でなくオブジェクト単位で行っても問題ない。

車は位置や姿勢が刻々と変化するので，オブジェクト単位での平行移動や回転を計算する必要がある。もちろん，地表や地面固定オブジェクトも見え方が変化するのだが，車はその中でさらに移動/回転する。そしてタイヤは車に追従しながらも車との相対位置や角度が変化するのである。

図4で分類したオブジェクトは上にあるものほど処理が単純で下にいくほど複雑になる。しかも，単純な処理ですむオブジェクトのほうが概してサイズが大きく，単純に処理することで節約できる処理量は大きい。

＊　＊　＊

システムの構造を整理していくと，車両挙動の処理は必然的にメインルーチンから分離することになる。コードの機種依存性も完全に排除される。すなわち，車両挙動を記述するために書いたコードが，いろいろなプラットフォーム上で使い回せるのだ(もちろん，C言語が使える処理系に限られるし，さらにシステムによっては浮動小数点を高速に扱えないので，そういう意味での制限は残る)。

これが，今回やったことの真の目的である。つまり，X68000/030に限らず，ほかの処理系に移植できるようにするための準備である。

図4 オブジェクト種別

	例	隠面処理(Zソート)	位置(平行移動)と姿勢(回転)の計算	上位物体との相対位置と相対姿勢の計算
水平面	地表, 路面	無	無	無
地面固定物体	壁, 橋	有	無	無
可動物体	車	有	有	無
下位構造物	タイヤ	有	有	有

DSPの可能性

Taki Yasushi 瀧 康史

DSPボードAWESOME-Xがようやく発売されました
この連載ではDSPのもたらす可能性を探っていくことにします
最初はDSPがどういう位置にあるのかから始めてみましょう

DSPがあったなら……

「DSPがあればできたのに……」

この台詞は，X68000ユーザにはよくいわれてきました。たとえば，ナムコのSYSTEM2基板*1の音楽をX68000で再現しようとしたとき，リッジレーサーを見て高速な画像演算を行いたいと思ったとき，ベクタフォントの展開が遅いとき……などなど。

いささか，過度の期待があるような気もしますが，その魔法の石，DSPはとうとうX68000用として接続されることになりました。そこで，DSPというものは，どういったものであるのか。いったい，どういうことができるのか。プログラムはどのように作ればよいのかなどを（私自身も勉強しつつ），数回に分けて解説していきたいと思います。

扱うDSPはTI(Texas Instruments)社のTMS320C26(以下C26)というものです。前世代のDSPに比べて，遥かに利用しやすくなってはいますが，汎用MPUに比べると，決してプログラミングはやさしいものではありません。したがって，読者ターゲットとしては，X680x0のアセンブラが，あらかじめ使えるぐらいのレベルでなくてはなりません。

今回は第1回ということで，DSP自体の説明です。先月，レビュー時にさらっと流したことを，少し込み入った話まで持ち込んで解説したいと思います。

*1 往年のナムコの高機能システム基板。同社の2D用システム基板の最高峰。アサルトで見せたスプライトの拡大縮小回転やワルキューレの伝説でのシンフォニックな音楽は当時のレベルから傑出した存在であった。

AWESOME-Xの構造

DSPというのは非常に広義な言葉です。もともとの，DSPは専用用途システムで，応用が利かない代わりに，専門演算は高速に行うことができるというものでした。したがって，音声には音声用，画像処理には画像用と分野別に利用するのが普通でした。専用化されたDSPは扱うメモリが少なかったり，ほとんどなかったり，プログラムが組めなかったりします。メモリがたったの64バイトで（64Kバイトではない），プログラムはリセットしたら起動し，絶えず64バイトの間を回り続けるだけのものもありました。

DSPの計算能力をできるだけ損なわずに，MPUが持っているメモリ管理機能などを付加する。つまり，専用化されたDSPをもっと一般化し，MPUのようにプログラムできたり，多量のメモリをアクセスできるようにするということが考えられました。現在のTI社の最新作は，確かTMS320C40だったはずです。このレベルまでいくと，OSが走りCコンパイラまで動いてしまいます。

ということで，DSPはピンからキリまであるのです。そしてどんどんとMPUとDSPの差が見えなくなってきています。コスト的にも，1チップ，500円で買えるものから，20万や30万を軽く超えるものまであります。分野が多岐にわたったDSPでも共通点はひとつ。つまり，ほとんど計算能力に重点をおいたものだということです。

C26はこの流れの中で，OSが動き，Cコンパイラも動くDSPです。メモリは最大64Kword(16ビット)×2本と，この世代では大きいとも小さいともいえないレベルです*2。個人的にはTI社でなく，モトローラのDSP56000系*3がよかったかなとも思いますが，考え方によってはC26もわりと，X68000にあっているのかもしれません。

さて，いくら速いDSPをつけても，ハードウェアアーキテクチャの上で，DSPがどこにあるかで，どの程度のことができるかが決まってきます。NeXTの初代は，カタログ値ではX68030と余り変わらないシステムですが，あちらはDisplayPostScript*4といった，いかにも重そうなことをしつつ，きちんと動いています。対してX68030のほうは，040turboを入れないと，満足にベクタフォントでワープロもできないぐらいの遅さです。この違いは，DSPが画面描画の近くにいるかどうかの違いです。

しかし，X68000に接続するとなると，拡張バスを通さなくてはなりませんから，このバス転送速度がかなりのネックになります。X68000の場合，拡張バスからバスマスタをするのは多少面倒なため，VRAMを直接アクセスする基板を作るには，相当の根性が必要です。MPUとのデータ交信が遅いと，DSPでせっかく計算をしても，計算結果を還元することができませんから，結局パフォーマンスアップは望めなくなります。

AWESOME-Xは，できそうにないことはスッパリと諦めて，できる範囲で努力した感じがする構成をしています。

*2 ほんとにピンからキリまでありすぎて，一概に大きいとも小さいともいえません。大容量のメモリを扱うMPUに比べたら，遥かに少ないですが，1Kwordに満たないメモリでFFTを1/44.1kHzの速度でリアルタイムにやっているのを見ると，2Kwordあれば十分かなという気もしてしまうのです。DSPの場合，高速な代わりに小容

量の高いSRAMを利用するのが一般的みたいです。画像を専門に処理するDSPでもない限り，あまり多くメモリを持ったシステムはそんなにはないみたいです。
＊3 MC68000のニーモニックに似たアセンブラを持ったもの。
＊4 PostScriptはもともとプリンタ用のページ記述言語で，いわばBASICのLINEやCIRCLEといったグラフィック命令で文字や絵を描いていくようなもの。それを画面表示にまで使用しようというのがDisplayPostScriptだ。基本的には3次ベジエ曲線でフォントなどを定義し，使用時に指定の大きさに拡大して表示する。これなら完全に表示画面と印刷物を同じイメージにできる。NeXTでは画面のすべてがDisplayPostScriptで記述，表示されている。

TMS320C26について

DSPを勉強するうえで注意しなくてはいけないことがいくつかあります。普通はMPUが暗黙に処理するので，気にならない部分ですが，DSPでは高速性を重要視するため当たり前ではなくなるからです。

たとえば一般の16ビットMPUでは，1word＝2bytes。つまり16ビットになります。本来，1wordというのは，データバス幅を示すのが本当ですから，DSPでの1wordはデータバス幅によって違います。TMS320C26はたまたま16ビットDSPなので，MC68000と違いがありませんが，24ビットDSPでは1wordは24ビットです。

また，MPUではデータバスの数にかかわらず，ひとつのアドレスに割り振られた値は必ず8ビットです。

わかりづらいでしょうから，具体的に説明しましょう。MC680x0なら，アドレス$00001000.wから順に，$aaab,$acadというデータが格納されているとき，$00001001.bの値は$abを指します。$00001002.bならば$acです。TMS320C26の場合アドレス$1000.wから順に$aaab,$acadと格納されているとき，$1000.wは$aaabですが，$1001.wは$acadになるのです。要するに，ひとつのアドレスに1wordが必ず入ります。だから記事中，1Kwordというのは2Kバイトのことを指します。単位をしっかり見るようにしてください。

次にメモリブロックの相違です。X68000では，メモリはプログラム用もデータ用も同じですが，TMS320C26ではプログラムメモリとデータメモリが違います＊5。つまり，同じアドレス上でデータとして見るメモリとプログラムとして見るメモリが違うということです。

TMS320C26の場合，オンチップデータRAMが1568wordあります。したがってAWESOME-Xは，プログラム用にSRAMが32Kword，データ用にX680x0から見えるRAMとして2Kword(ウェイトあり)，X680x0からは見えないRAMとして1568word(ウェイトなし)のメモリがあるわけです。

DSPが一般にプログラムがしづらいといわれるのはこれ以外にも原因があります。たとえば，メモリを節約するために，スタックをできるだけ利用せずに行うこと。つまり，プログラムをあまりサブルーチン化せずに，かつ，がむしゃらに展開して長くしないことなどなど。これらはDSPのコーディングをするときに細かく話を進めましょう。

現状で勉強しておかねばならない知識はこれぐらいです。

＊5 X680x0では利用されていませんが，MC68000は，プログラム用，データ用，スタック用と個々にアドレスを分けることができます。

ブロックダイアグラム

ではブロックダイアグラムを見てください。これは，AWESOME-Xのマニュアルに書かれた資料を見つつ，適当に私が作ったものです。だから，多少，おかしなところがあるかもしれません。おかしなところが見つかり次第，連載中で訂正をしていきましょう。

では具体的に説明を始めます。

●ステータス

まず，MPU-MC680x0と，DSP-TMS320C26の間を結ぶラインが，ステータス用の信号ラインです。この信号ラインは，お互いのステータスを与えあうための，いわばコントロール用のラインです（ブロックダイアグラムはこう書いてもいいのかな？)。

具体的には，以下のことができます。

○MC680x0→TMS320C26

アドレス$ec8000.bを利用して，コントロールを行います。このポートによって，バンクRAM0,1,2の切り換え，DSPに対する割り込み要求，リセット，ホールトができます。また，2ビットのフラグポートが用意されていて，自由に利用することができます。このポートは，DSP側でもX680x0側でも読み書きができるので，データ転送の際のハンドシェイクに主に利用されます。

○TMS320C26→MC680x0

TMS320C26からはMC680x0に対して割り込みを送ったり，2ビットのフラグポートを利用してハンドシェイクしたりできます。

また，DSPがBUSY中であるかどうか，

図1 AWESOME-X's Block Diagram

DSPがホールトされているかを，MC680x0は$ec800.bを通して知ることができます。

●多量のデータを送る場合

多量のデータを送るときには，RAMを利用します。MC680x0からは$ec8000を利用して，0～2のバンクを切り換えます。このバンクは68000上のアドレス，$EC0000～$EC7FFまで，32Kバイト分マッピングされます。

バンク0はSRAMで，DSP側から見ると，プログラムメモリの前半，$0000.w～$3FFF.Wの16Kwordを指します。このメモリを68000がアクセスするときは，DSPをホールドせねばなりません。

バンク1はSRAMで，DSP側から見ると，プログラムメモリの後半，$4000.W～$7FFF.Wの16Kwordを指します。これもDSPをホールドする必要があります。

バンク2はDPRAM（デュアルポートRAM)です。ただしメモリは4Kバイトしかないので，680x0から見たら，$ec0000～$ec01ffまでとなります。DSP側から見ると，データメモリの$0000～$00FFです。このメモリはDPRAMですから，DSPが動作中にも68000から読み書き可能です。DSPがこのメモリを利用して演算をするなら，演算の様子をリアルタイムで68000から見ることができます。

また，このDPRAMにはX680x0のDMA:63450でアクセスできるそうです(開発者からの伝聞)。

●DAI/RS232Cを利用する場合

DAIつまりPCMは，DSPのシリアル出力を利用します。ブロックダイアグラムから見てもわかるとおり，68000からは見えません。これはRS232Cも同等です。どちらもシリアル出力なので，同様のプログラムを利用します。

●X680x0のハードウェアを利用する場合

X680x0のハードウェアはMC680x0のメインバス（図中の左側の大きな矢印）の下につながっています。ブロックダイアグラムを見ればわかりますが，DSPとMPUは同一バス上にはいないため，直接，X680x0の周辺機器を操作することはできません。つまり，DSPがFM音源チップを操作したり，X680x0を操作したりすることはできないということです。

バスマスタ*6をしないと，このような不利益がありますが，その代わり，2つのプロセッサは完全に別々に並列進行することができるという利点があります。

＊　＊　＊

以上のように，ブロックダイアグラムと，そのハードウェアの相関関係から，DSPでのプログラムの動作の仕方がある程度定まります。

利用の基本は，

1) DSP HOLDフラグを立てて，DSPを停止する
2) アクセスバンクをSRAMにし，プログラムをDSPメモリにロードする
3) アクセスバンクをDPRAMに指定する
4) DSP HOLDを解除する
5) DSPスタート命令を出す

このようになります。必要があるなら，これをうまく繰り返して，データとプログラムをX680x0側からロードしたりします。

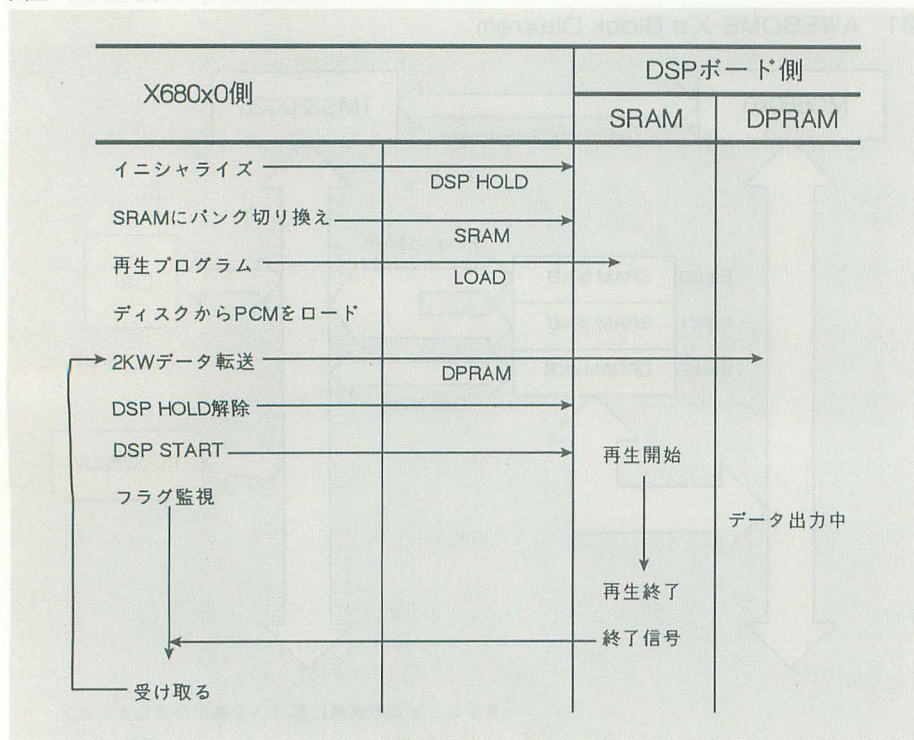
プログラム中，DSP側から割り込みがかかった場合，

a) DSPから割り込みがかかる
b) X680x0が割り込み処理を行う
c) X680x0は割り込み終了処理をDSPに伝える

以上のようになります。DPRAMがマッピングされている間は，2つのプロセッサは連動しているので，プログラムをうまく並列動作させるのが，高速動作プログラム作成ポイントといえるでしょう。

＊6 MPUのメインバスに対して，周辺機器が主導権を握ること（マスタ)。DSPがバスマスタするということは，MPU:MC680x0に代わって，DSP:TMS320C26がX680x0のバス所有権を一時的に持つことを指す。別にSCSIの規格ではない。

図2 DAIの出力タイムチャート

X680x0側
DSPボード側
SRAM
DPRAM
イニシャライズ
DSP HOLD
SRAMにバンク切り換え
SRAM
再生プログラム
LOAD
ディスクからPCMをロード
2KWデータ転送
DPRAM
DSP HOLD解除
DSP START
再生開始
フラグ監視
データ出力中
再生終了
終了信号
受け取る

可能性

先月のレビューでは可能性を漠然とあげましたが，今回はその裏付けを考えつつ，どのようなプログラムが作れるか考えていきます。

●DAIを利用した音源ドライバ

DAIでPCMを再生するときには，DPRAMを酷使しなくてはなりません。付属のソフトは，次のような手順で再生しています。

1) 基本的手順に基づいて，DPRAM:2Kwordをループ再生するDSPのプログラムをプログラムメモリにロードする。このプログラムは，停止命令が出されるまで，DPRAMを連続して再生する
2) DPRAMにバンクを切り換える
3) ディスクからPCMデータを，X680x0メインメモリにロードする
4) 4Kword,DPRAMにメモリを転送する。
5-1) DSPにスタート命令を出す。DSPは

DPRAMのデータを48kHzステレオ16ビットで再生をする。DSPはデータを再生し終えたら，フラグを立てる。
5-2) この間，メインCPUはフラグを監視し続ける
6) 4)に戻る

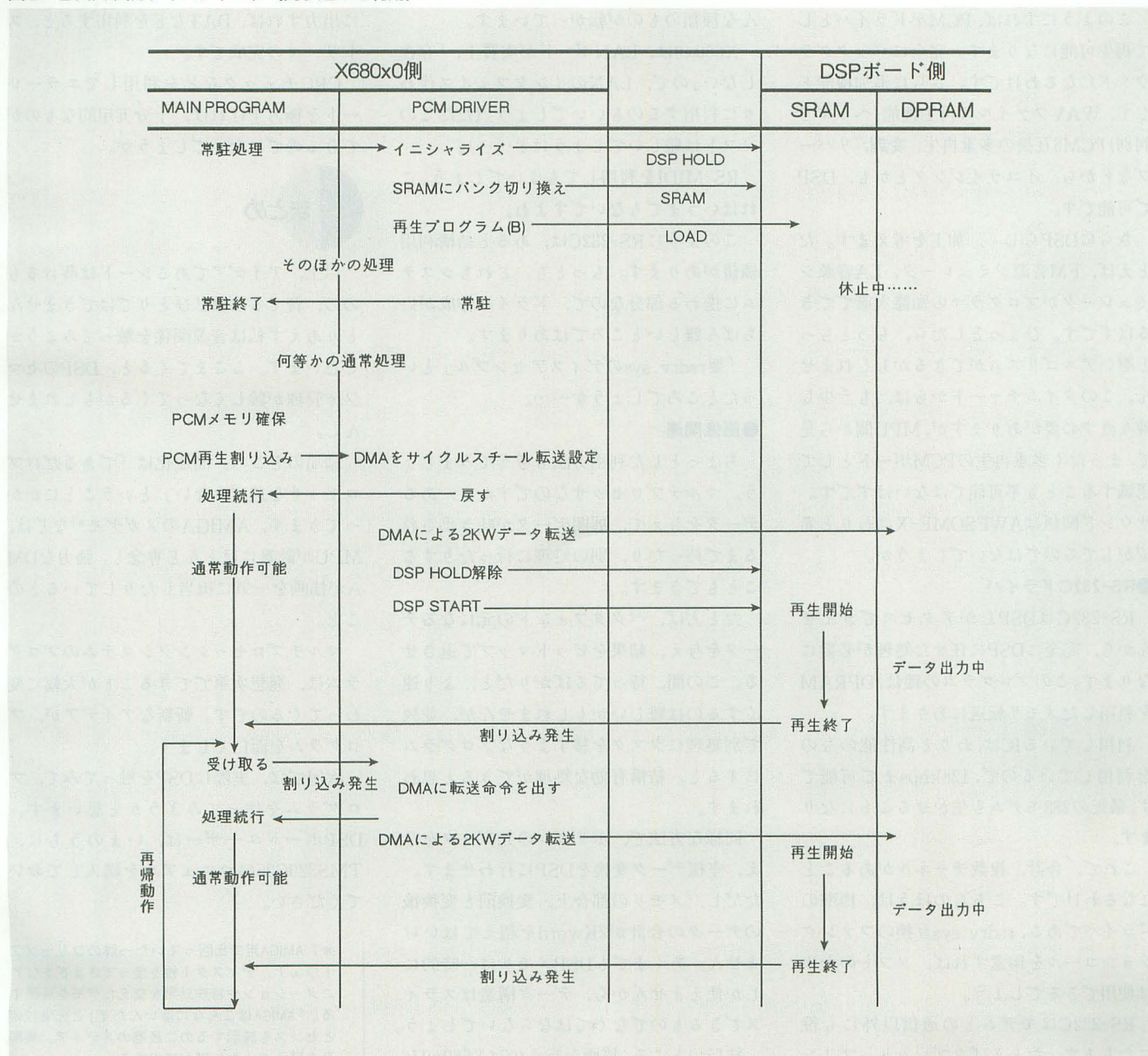
もう少し，賢い作り方をしていたような気がしますが，だいたいはこんなところです。タイムチャートでは基本的手順を踏まえて，より細かく書いてあります。参考にしてください。

この方法なら非常に簡単にプログラムを作成できますが，再生中，MPUは完全に停止状態になるため，あまり効率的とはいえません。実際にDSPを利用する場合，割り込みを酷使してプログラムを書く必要があります。

ソフト的にはX68000の世界で事実上標準であるPCM8互換コールを持たせれば，いままでの資産を生かすことができます。こうすれば，Z-MUSICでのPCMの再生を，DSPを利用して再生することもできます。

実際には，次のような処理をすればできるはずです。図3のタイムチャートを見てください。プログラムは，PCM-DRIVERに関する部分と，そうでない部分があります。DRIVERに関する部分は，割り込みによって操作されます。常駐処理までは，通常利用とあまり変わりません。

1) プログラム中PCMメモリが確保される
2) PCM再生割り込みがかかる
3) MPUはDMAに対して，PCMメモリから2Kword分，データをDPRAMに転送するように命令する
4) DRIVERの中で，DSP HOLDを解除し，DSP STARTを実行する
5) 通常処理ルーチンに戻る
6) この間，DSPは再生を続ける。MPUとDSPは完全に非同期動作を続けている
7) DSPはMPUに対して，再生終了の割り込みを発行
8) MPUはこれを受け，DMAに再転送の

図3 DAIの出力タイムチャート（割り込みを利用）

命令をする

9-1) MPUは通常処理を続ける

9-2) DMAはデータを転送する

9-3) DSPはデータを再生する

10) 7)に戻る

このような処理になるはずです。MPUは途中で割り込まれて，DMAに指令を出すだけですから，非常に軽くなるはずです。また，9)の間，3つのICが並列に動くことになります。

7)～9)の間は1/48kHzの時間より短くなくてはなりません。そうでないと，この間，空きができてしまいます。DMAが利用されているなら，MPUで転送しますが，この速度は十分間にあうでしょう。

このようにすれば，PCMをドライバとして再生可能になります。完全にバックグラウンドになるわけです。さらに追加機能として，WAVファイルの再生機能(ヘッダで判別)PCM8互換の多重再生，変調，リバーブなどから，イコライジングとかも，DSPで可能です。

さらにDSPらしく，加工を考えます。たとえば，FM音源シミュレータ，LA音源シミュレータがプログラマの知識次第でできるはずです。ひょっとしたら，もっともっと凄いアルゴリズムができるかもしれません。このタイムチャートからは，もう少し練り直す必要がありますが，MPU側から見て，まったく多重再生のPCMボードとして認識することも不可能ではないはずです。サウンド関係はAWESOME-Xはわりと希望がもてるのではないでしょうか。

●RS-232Cドライバ

RS-232CはDSPしかアクセスできませんから，完全にDSPに任せた処理が必要になります。このプログラムの鍵は，DPRAMを利用したメモリ転送にあります。

利用しているICは，わりと高性能のものを利用しているので，128kbpsまで可能です。最近の288モデムも生かせることになります。

これで，合計，複数チャネルがあることになるわけです。こちらのほうは，標準のドライバである，rsdrv.sys互換のファンクションコールを用意すれば，ソフトの大半は使用できるでしょう。

RS-232Cはモデムとの通信以外にも役にたちます。たとえばタブレット，プリンタ，RS-MIDIなど。タブレットなどはたいした転送を行いませんし，MATIERなどは直接アクセスしているでしょうから，標準のRSポートを利用したとしても増設の2chで通信することができます。

また，プリンタなどもRS-232Cのものがあります。たとえば，Macintosh系ではRS-422でプリンタを制御していますし。ドライバは現状ではありませんが，カラーインクジェットプリンタをRSへ，レーザープリンタをパラレルへつなぐことができます。X680x0では，AdobeのポストスクリプトプリンタがRS-232C接続ですしね……。

マウスやトラックボールなども利用できます。PC/AT機のマウスはシリアルでいろんな種類のものが転がっています。

X680x0は，LANボードが実質上，「存在しない」ので，LANのインタフェイス代わりに利用するのもいいでしょう（ただこのソフトは難しいでしょうけど……)。

RS-MIDIを利用してもよいでしょう。これはいうまでもないですよね。

このようにRS-232Cは，あると結構利用価値があります。もっとも，どれもシステムに携わる部分なので，ドライバ作成がいちばん難しいところではあります。

「要rsdrv.sysのディスアセンブル」といったところでしょうか……。

●画像関連

ちょっとした利用方法も考えてみましょう。マルチプロセッサなのですから，あるデータを与えて，展開データが吐き出されるまで待ったり，別の処理に行ったりすることもできます。

たとえば，ベクタフォントの元になるデータを与え，結果をビットマップで返させる。この間，待ってるばかりだと，より速くするのは難しいかもしれませんが，並列で別処理にタスクを移すようなプログラムにすると，結構有効な処理ができると思われます。

同様な方法で，ポリゴンの元データを与え，座標データ変換をDSPに行わせます。ただし，メモリの都合上，変換前と変換後のデータの合計が2Kwordを超えてはいけません。あくまでもDSPメモリは一時的にしか使えませんから，データ構造はスライスできるものでなくてはならないでしょう。

結局のところ，描画を行うのはX680x0なので，リアルタイム処理にはあまり向いてるとはいえませんが，プログラムのタイムチャートを作り，うまく2つのプロセッサを動かせるようにすれば，それなりの恩恵は得られるかもしれません。

たとえばSLASHなどは，頂点演算をすべてDSPにやらせるようにしたら，それなりの高速化が望めるのではないでしょうか。

●DAIを利用したDATストリーマ

DAIを利用したDATストリーマを作れます。データメモリは2Kword＋1568wordですから，HDDのデータを4Kバイト単位で分割し，エラーチェック用のデータを1568wordに埋め込むことができるでしょう。この埋め込んだデータをあわせてDAIに出力すれば，DATなどを利用すると，ストリーマの完成です。

CRCチェックなどを利用してエラーレートを極力下げれば，十分実用的なものができるのではないでしょうか。

まとめ

以上，アイデアであるシードは蒔けるものの，育てるのは私ひとりではできません。とりあえず私は音源関係を触ってみようかと思います。ここまでくると，DSPのセマフォ管理が難しくなってくるかもしれませんし。

結局のところ，高速化は「できるだけプロセッサを遊ばせない」ということにかかってきます。AMIGAのメガデモ*7などは，MPUが計算にほとんど専念し，強力なDMAが描画を一気に担当したりしているとのこと。

マルチプロセッシングシステムのプログラムは，発想次第でできることが大幅に変わってくるのです。斬新なアイデアが，プログラムを面白くします。

次回では，実際にDSPを触ってみて，プログラムを作ってみようかと思います。DSPボードユーザーは，いまのうちに，TMS320C25のマニュアルを購入しておいてください。

*7 AMIGA用で出回っていた一群のフリーソフトウェア。ディスク1枚を使ってさまざまなアニメーションや特殊効果を交えたデモを展開する。「AMIGAはこんなに凄いんだぞ」と先端技術とセンスを誇示するのに最適のメディア。実際，目を疑うような処理が続出する。

ローテク工作実験室　第10回

D/Aコンバータの制作

Taki Yasushi 瀧　康史

DSPボード以外にも一般化しつつある光オーディオ出力。デジタル光信号をアナログの電気信号に変えるにはコンバータが必要です。誰がやってもだいたい作り方は一緒……ということで今回は市販キットの紹介です。

人様の座敷に上がってみて

先日の編集後記にて,「そろそろ旅立ち先を考えたい……」なんていっていたんですけど……とりあえずは手持ちのH98にウィンドウアクセラレータをつけて，メモリをあわせて20Mバイト程度まで増設してみました。初めて「まとも」に触るMS-WINDOWSでしたけれど……まあ，慣れないものを触る楽しさはあって，結構遊べることは遊べました。

だけど，あのウィンドウシステム，なんでボタンが2つあるんだろうって考えてしまいます。「ああ，このウィンドウシステムは左ボタンでプルダウンメニューが出るのか」と思いきや，アプリによっては右でポップアップメニューが出てきます。まあWINDOWS95が出るまで待ちますか。今度はちゃんとマンマシンインタフェイスを考えて作ってるっていいますし。

同じくして，Macintoshを人様の座敷にて使ってみました。ああ，Macintoshって1ボタンで十分自然だなと思います。しかし，Macintoshはマルチタスクですけど，シングルジョブなウィンドウシステムっぽいですね。初めて使っても結構使えるという「代償」なんでしょうけど。画面全体で「私はいまマックドローを使ってるよ！」と訴えてます。

ウィンドウシステムというのは，複数のタスク（コンピュータでいう仕事の単位）を同時に使えるからよいのじゃなくて，複数のジョブ（人間の仕事単位）を同時に行えるからいいんだと「私は」信じてます。あっち行ったりこっち行ったり。

わがままといえばそうなんですけど，この場合，マルチジョブというよりも，ジョブリンクスって感じかな。関連のある複数の仕事を同時に行うというか。とはいえ，こういうことが互いにできるようになると，初心者お断りな環境ができそうではあります。でもSX-WINDOWはそうしながら使っています。いまは裏でQuTERMが走っていて，ワークステーションにアクセスしてますし。そういうわけでMacintoshは初心者をよく意識したものですが，私にはあわないと感じました。

で，今度はX WINDOWを利用してみました。悪くないんですけど，私には3つもボタンはいりません。それから，必ずコンソールウィンドウが携わってくるという感じがあるんですが，これはカスタマイズの問題かなあ……こういう概念がいまいち。複数のウィンドウ間でタスク間通信をするというのは悪くないとは思いますけれど。まあ，X WINDOWシステムは，カスタマイズによっていくらでも変わるみたいですから，もう少し使ってみるつもりです。しかし，乗り換えるとするなら，マシンが高いのが最大の欠点。これを「まともに」動かすためのハードウェアが100万円では買えないところが痛いところです。

最後に再びSX-WINDOWに戻ってきました。やっぱりマウスボタンは2つがちょうどいいなあと思うこの頃。でも，ウィンドウアクセラレータがほしいなあ。気にいってはいるウィンドウシステムだけれど，そろそろ愛用のマシンの機能に限界が見えてきました。6万色だとマッハバンドやら，ディザパターンが気になってしまってしょうがないですし。16色なら1024×768で快適ですけれど，多色ではGRWの512×512の中しかダメ。1677万色で1280×960ドットぐらいほしいのが実情です。メモリだって少ないですね。X-DTPでイメージをちょっと多く貼るとすぐメモリがたりなくなってしまいます。

ところで，PC/AT機が安いなんていったの誰？　ちょっと気のきいたシステムを作ろうとすると，すぐに50万円超えてしまうんですけど。ほしいのは，X68030で手が届かない部分。カスみたいな奴は安いということは認めるけど，そんなのはいりません。ちょうどほしいのはものすごく高いのです。少なくともX68030よりも使えるシステムにしないと，引っ越し作業は意味がないと思うのに，ソフトまであわせたら100万円を優に超えちゃいます。ラインナップがたくさんあるのはいいんですけれど。

もう少しSX-WINDOW on X68040に，小回りのきいたローテクで我慢するのがいいのかな。そしたら，そのうち，シャープがなんか出してくれるでしょう。

キットだときっと楽

つまらん……。

たとえDAI付きの高品質DSPボードを持っていても，デジタルアンプやDAT，MDなどがないと，オーディオ出力にはできません。だからといって，デジタルアンプは高いし，DAT，MDは録音モードにしないとオーディオ出力から出力されません。

「うちのDATは同軸だからつなげないよ。もっと簡単にDSPボードのPCMをオーディオ出力する方法はないの？」

そういう方も多いと思います。おあつらえ向きのキットが秋月電子から出てるんですよね。値段はあわせて5,000円以下。通販も行っているので，地方の方でも購入可能ときています。

この手の回路はお決まりというものがあって，誰が作っても似たようなものになります。そこで，私がオリジナルなものをわざわざ作るよりも，あるものを利用したほうが絶対よいし便利なのでこれを使ってみることにしましょう。

DAIの規格について

DAI（Digital Audio Interface）という規格は，伝送ラインに同軸，BNCを求めて

います。同軸というのは，普通のピンジャックに信号線,GND線をあわせたもの。BNCはケーブルとコネクタが変わっただけであとは同じです。もうひとつ，オプティカルケーブル(光ファイバ通信)があります。プロ仕様であるなら，端子は3端子。たぶん，信号のプラスマイナスとGNDだと思いますが。

BNCを使うことはそうはありませんから，一般にはDAIがついたオーディオ機器は，オプティカルと同軸の2種類に分けられます。デジタルアンプはさすがにどちらでもつけられますけど，普通のオーディオ機器は，どちらか一方しかついていません。

そのくせ,この同軸↔オプティカルの相互コンバータなるものが必要になるはずなのですが，私が調べた限りでは1社からしか出てなくて，それもセレクタに内蔵されていて，1万円以上するものでした。

最近私が買った小型（といっても持ち歩きたくなるほどには小さくない)の安物DATは，シャープ製のもので29,800円でした。必要最低限の機能しかついてなく，DAIは同軸です。どうやらオプティカルのDAIを装備したものは，一般には大きめのシステムしかないみたいです。

AWESOME-Xが同軸にしなかったのは(推測ですが）理由があります。結果からいうなら，それはノイズをなくすためです。

単なるPCMボードを作るならば,ボード上に,PCMのD/Aコンバータを内蔵すればそれで終わりです。PC-9801のサウンドボードなどもそうしています。しかしながら,D/Aをボード上に搭載してしまうと，コンピュータのノイズが乗ってしまいます。これだといただけないということで，DAIにするのです。

しかしながら，たとえDAIにしたとしても，同軸の場合はあくまで「電送」です。したがって，最終的には電気的に接続されてしまいます。電気的にGNDラインが電送されているということは，その分，完全にノイズをシャットアウトしたことにはなりません。

オプティカルケーブルで接続すると，電気的には完全に絶縁されるため，コンピュータのノイズは，D/Aコンバータまで流れないことになります。それゆえ，オプティカルケーブルを利用しているのだと思われます。

回路構成

装置の回路構成を簡単に述べてみましょう。

信号はオプティカルで入ってきますから，これを電気信号に直します。電気信号に直すためには，東芝TORX178Aという光受信モジュールというものを利用します。この出力は同軸DAIと同じものです。

電送はシリアルなので，DAI復調ICを利用して，D/Aコンバータに入力するための別の形式に変換します。この処理をするのがPD0052というICです。

これによって3つの信号が出力されます。BCK(ベースクロック)，L/R信号，DATA信号の3種類です。データはリトルエンディアンで転送され，L/RがHighのときに送られるDATA信号が，左の16bitデータ。L/RがLowのときに送られるデータが右のデータです。

デジタルオーディオ機器には，32kHz，44.1kHz，48kHzの3種がありますが，この速度はBCKで決まります。ものによっては18ビットのものもあります。

どうやらこのBCK，L/R，DATA信号という形式は，どのメーカーのICでもほぼ同じらしく，デジタルオーディオ装置は，この信号にしてデータをやりとりしています。変調ICを利用すれば,手持ちのCD-ROMなんかにDAIをつけることだって可能です。

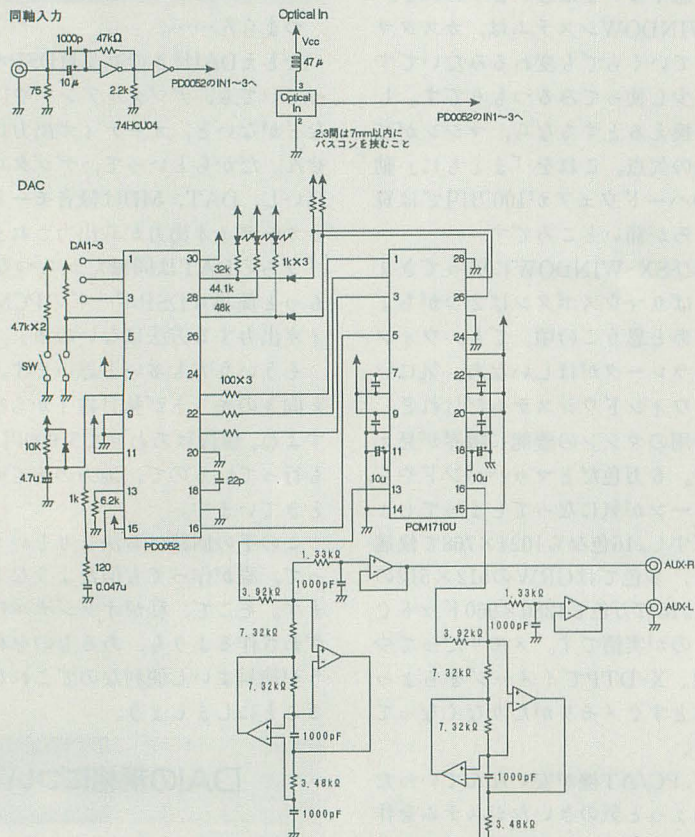
この3つの信号を作ってやれば，簡単に

キットが発売停止してたらどうしよう……

ということで，私が別の実験で作成した回路のほうを掲載することにします。図がその回路で，PCM1710Uを利用しています。

このPCM1710UというD/Aコンバータは，8倍オーバーサンプリング,16/20ビットのデジタルフィルタを内蔵した，わりと高性能のICです。PD0052も20ビットモードに対応していて，このICも20ビットモードに対応しているので，20ビット出力ができないことはないんですが，残念ながらPCM1710Uは，コントロール入力をシリアルで送らないと，20ビットモードになりません。このコントローラを作るために，たいそうな回路を作らねばならないため，回路を単純化させるために，このサンプルでは16ビットモードで利用しています。残念。

PCM1710UもPD0052もわりと小型のICです。電源周りの回路はスペースの都合で割愛させていただきました。レギュレータを利用して，5Vを作ってください。その際，PCM1710UとPD0052のレギュレータを別にすると，音質が多少変わる……かもしれません。

実は自作PCMボードにこのICを利用しようかな……と考えていたのですけど。最近は忙しい＆お小遣いがないということで，ボードは進まなくなっちゃっています。すみません。

D/Aボードを作ることも可能です。PC-9801用のサウンドボードである86ボードは，この信号を，FIFOメモリ→YAMAHA YM3434(8倍18ビット化デジタルフィルタ)→B.B PCM61P(18ビットD/Aコンバータ)→LPF→AUXという具合にPCM出力しています。

今回のキットの中に入っているD/Aコンバータは，SANYOのLC7883という，デジタルフィルタ内蔵の16bitD/Aコンバータです。しかもL/R2ch搭載。小さくて，安価なフラットパッケージのICです。

実は，このキットを見つける前に，同じ処理をするボードを作ってみました。そのときの回路は，PD0052→YM3434→PCM61P(18ビットPCM)×2chという構成。PCM61Pは小さなDIPですが，実は右，左の2ch分，用意しなくてはなりませんでした。一応力技で，ひとつのICをもってステレオ出力する方法がありましたが，その分OP-Ampが増えてしまうという有様。

それ以外にも，PCM58P(61Pとほぼ同じ性能で異常にデカいIC)，PCM56P(61Pとピン互換だけど16ビット)という具合に，いろいろ試してみました。YM3434＋PCM58P×2chのICセットなんて回路だけでものすごいサイズになります。18ビットだから多少音がいいかな？　とも思ったのですが，LPF段階で工夫したほうが，遥かに音質が変わるんですよね。

もっと高性能のICで，B.B PCM1710Uという8倍デジタルフィルタ，2ch内蔵のものがありまして，これも利用してみました。これはさすがに音がしっとりとするというか，なんというか，多少変わったかなという感じ。ただし調子のよいヘッドフォンでよく聞くと……ですけど。これは，8,000円もします。

と，ここまで考えると，このSANYOのLC7883はなかなかいいものかもしれません。

最後段でLPFをかまします。キットはNJM4580という2ch内蔵で，単電源動作で，オーディオ用，ノイズの少ないわりとよさげなOP-Ampです。これで出力を調整して終わり。

回路構成としては単純なものですが，これだけで十分な動作を果たしてくれます。

キットを利用時の注意

このキットは「プリント基板」を作るキットなので，そのほかの部品はセットになっていません。これ以外に，ハコ，端子，電源などが必要になります。

ハコはできれば，金属製のハコを用意し，回路のGNDとフレームを接続してください。ノイズ対策になります。もっとも，金属ハコとプラスティックハコでは，加工のしやすさと値段が全然違います。私は加工の面倒くささに耐えかねて，結局，プラスティックケースで作ってしまいました。基板サイズが5cm×5cmですから，ハコは十分なサイズを用意してください。

組みあげた基板

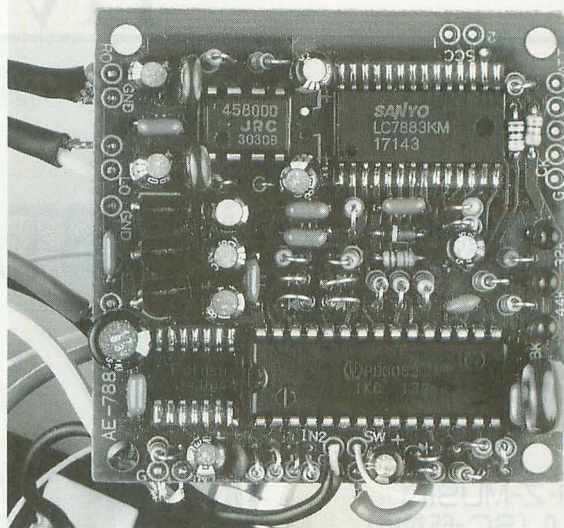
端子はいくつか必要です。まずはオプティカル用の端子。同じく秋月電子にて送受信セットの700円で購入。ただし，今回のキットでは受信だけしか利用しません。

キットの入力は2系統あるので，オプティカルのほかに，同軸もつけられます。その場合，切り替え用のスイッチと，同軸用のピンコネクタが必要です。AUX端子と同じものですね。これは千石電商にて購入。同時にAUX用に赤と白のものを手に入れるとよいでしょう。写真では，同軸とオプティカルを切り替えにしました。もちろん両方オプティカルにもできます。適当にどうぞ。

電源は12Vのものが必要です。アダプタが秋月にて300円で置いてあります。12Vだからといって，本体から取ってしまうと，ノイズをまた乗せてしまうことになります。いくらキットの回路上にて，レギュレータを利用してノイズを除去したとはいえ，それでも別電源のほうがいいです。

アダプタの端子は千石電商にて，100円で売っていますが，似たサイズのものがあるので，2.1φのものを間違いなく買ってきましょう。もちろん，オプティカルケーブルも別クチで購入してください。

製作については，キットのマニュアルにほとんど掲載されていますから，特に困ることはないでしょう。ただし，秋月のキットは，「電子工作ができる人」向けのキットですから，抵抗が読めたり，ハンダ付けができたり(フラットが2つあります)，コンデンサの容量が読めることが必要です。

また，キットの中では10～100pFのように，具体的な値をきっちり書いてないものがあります。キットの中をくまなく探し，値が指定されているものを除いて，余っているものから適当そうなものを利用してください。たぶん，キットごとに違うんでしょう。

それから，キットを購入してきたら，まず最初に部品のチェックをしてください。これは常識です。

まとめ

秋月電子には，そのほかにもいろんなキットが売られています。別に私は秋月電子の回し者ではないんですが。足を運ぶことができる人なら，出歩いてみるのもよいでしょう。RGBから複合映像信号やY/C分離信号に変換するものなど，いろいろ便利なものがあります。

SRAMボードですけど，資金と暇がなくてなかなか作成できません。この手のものは，1週間ぐらい「マジ！」になればできるものなんですが，マジ以外にもお金がいるんですよね。

身体が2つも3つもほしいこの頃。多趣味なのが問題なのかもしれませんけど。

部品表

○D/Aコンバータキット
- fs=32/44.1/48kHz対応DAI搭載
- DAC：LC7883，DAI：PD0052，BUFF：NJM4580DD
- DAIはIVCO，IPLL構成で384fsにロック
  同軸入力，オプションでオプティカル
- DACは16bitダイナミックレベルシフト変換，8倍オーバーサンプリング
- 専用基板：50×50mm(ガラエポ両面)
- 電源：要DC12V，100mA以下
  (3端子レギュレータ内蔵)

3,600円

○DAI用，光送受信モジュール
- TOTX178/TORX178ペア

700円

問い合わせ先：
(有)秋月電子通商
☎03-3251-1779(12:00～18:30)

Oh!X LIVE in '95

X68000・Z-MUSIC ver.2.0（SC-55対応）
「ファイナルファンタジーV」より ©1993スクウェア
暁の戦士
Tanabe Masanori 田辺 正則

X68000・Z-MUSIC ver.2.0（SC-55対応）
「ドラゴンスレイヤーVI」より ©日本ファルコム
STAR GAZER II
Kurachi Kazuhiro 倉知 和弘

X68000・Z-MUSIC ver.2.0（SC-55対応）
SAY ANYTHING
Tsukamoto Takehiko 塚本 岳彦

X68000・Z-MUSIC ver.2.0（SC-55対応）
WAIT FOR SLEEP
Chikira Kazuhiko 千喜良 和彦

X68000・Z-MUSIC ver.2.0用
「ときめきメモリアル」より ©KONAMI
告白
Sasaki Tsugutomo 佐々木 嗣朋

今月のLIVE inは夏休みスペシャル（9月号だけど……）ということで一挙に5曲掲載です。手軽に楽しめる短めの曲から，怒濤の大作まで取り揃えてありますのでぜひ入力して聞いてみてください。

エクスデス城にて

Oh!X LIVEでは「ファイナルファンタジ」ーシリーズの曲とコナミの「グラディウス」系の曲は激戦区です。よって掲載された作品はその激戦を勝ち抜いたモノということになります。今回，お届けするのはファイナルファンタジーVより「暁の戦士」です。

データ制作者の田辺君はゲームをプレイしていてエクスデス城にたどり着いたときこの曲を聞き，あまりのカッコよさに感動してしまったそうです。そしてゲームを放り出してこの曲づくりに没頭してしまったとのことです。それだけの思い入れがあるからこそこれだけの完成度を達成できたんだと思います。

曲は勇ましいマーチ調です。ティンパニのリズムに乗ってブラス隊が雄大なメロディを奏で，そしてバックのストリングス隊がこのブラスメロディに美しいハーモニーを与えつつオブリガードを展開します。

この曲の聞きどころはなんといってもティンパニです。ティンパニは単なるリズム楽器として軽視されがちですが実は交響曲では最重要の打楽器です。一定の音高を持つ唯一の太鼓でバスを強調し，ときには低音部ハーモニーの重要な役割を果たします。このデータでもあたかもメロディ楽器のように音の高低を意識してシーケンスされていますがこれは決してデタラメではなく意味があるのです。その辺りを理解して入力し，聞いてみましょう。

演奏にはGS音源が必要です。編集室でSC-55/SC-55mkII/SC-88での正常な演奏を確認しています。

夜空を見上げよう

ウィンダムヒルを思わせる静かなピアノ曲をお届けします。

暑い夏を駆け抜ける一陣の涼しげな風，そんなイメージを彷彿とさせるこの曲「STAR GAZER2」は，本誌ライターの瀧氏絶賛のCD「プレプリマII」（キングレコード）に収録されています。この「プレプリマ」シリーズはファルコムのゲームミュージックのアレンジバージョンCDシリーズなのですが，他社の同種のモノとは一線を画した存在です。というのも原曲をアレンジした，というよりは原曲にインスパイアされて新たに作曲したといった感じの曲が多く，よい意味で原曲とはかけ離れているからです。となるとゲームミュージックというサブタイトルを持つことがどれほどの意味があるのか疑問ですが，とにかく音楽的には優れたものが多く収められているのでぜひ一度ご一聴されたし。

さて，曲データのほうはピアノ曲ということでトラック数は3と少なめです。リストの長さも短めですが，右手パート，左手パート，というトラック振り分け，つまり奏者の視点の構成を取っており本格指向なデータといえます。テンポやベロシティの変化などのさりげない情緒演出も見事，そしてエフェクトもエキサイターがかった感じのピアノサウンド向きセッティングでミキシングバランスも満点です。バックにかすかに聞こえる波の効果音も雰囲気でています。

演奏にはGS音源が必要です。編集室でSC-55/SC-55mkII/SC-88での正常な演奏を確認しました。

いい忘れるところでしたがこの曲の原曲は「ドラゴンスレイヤーVI」のゲームオーバーのテーマ(だそう)です。

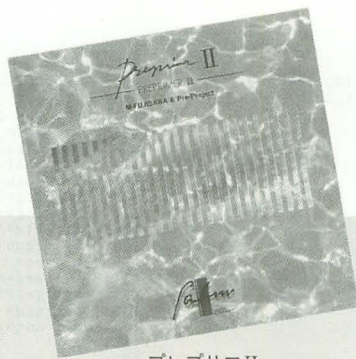
without any words.

X JAPANの曲ですが，バラードです。ハードロックバンドには必ずといっていいほどこの手の感動的なバラードがありますが，この「SAY ANYTHING」も大変メロディの美しい曲です。

壮大なストリングス隊のイントロがフェードインで始まり，そしてまたすぐにフェードアウト……と思いきやピアノのリズミックな伴奏が被り，さりげなくオルガンリードが歌い始める……。聞き手を摑むワザを心得ていますね。途中のギターソロも素晴らしいしメロディの音色の選択もいいと思いますが，聞きどころはバックのピアノです。普通だとバックパートは案外疎かになるものなのですが塚本君はこれを丹念にシーケンスし実に気持ちのこもった演奏を作り出しています。この曲の「縁の下の力持ち」的存在です。

演奏にはGS音源が必要です。なお，編集室でSC-55/SC-55mkII/SC-88による正常な演奏を確認しました。

目覚めと夢の硲に

次もヘビメタバンドの曲ですが，バラードでもないラプソディとはちがう，レクイエムなのかどうかもわからない，ちょっと線引きの難しい技巧的な曲をお届けします。いわずとしれたドリームシアターの「WAIT FOR SLEEP」という曲です。

プレプリマII

ときめきメモリアル

この曲はピアノのアルペジオとストリングスの美しくも緊張感のあるハーモニーから始まります。一見単調ですが生き物のように刻々と様相を変え，聞き手の予測を裏切って展開していきます。いわゆる変拍子の曲で聞くほうにもそれなりの「構え」が必要な曲かもしれません。千喜良君はこの曲をすべて聞き取りで作り上げげたというのですから凄いものですね。

演奏にはGS音源が必要です。編集部でSC-55/SC-55mkII/SC-88での正常な演奏を確認しました。

恋をしよう

PCエンジンで絶大な人気を誇っている恋愛シミュレーション「ときめきメモリアル」の準エンディングテーマともいえる「告白」のテーマを内蔵音源でお届けします。

この曲の旋律はこのゲームの主題ともいえるメロディで，ゲームのいろいろなシーンでその時々にあったアレンジで登場します。「告白」ではこのメロディがオルゴールの音色で，なんとも切ないハーモーニーを奏でます。

佐々木君はこの曲を内蔵FM音源のみで制作してくれました。使用音色も1種類のみでリストも100行程度です。AD PCM音源も使用していないのでリストを打ち込んだら，

A>ZMUSIC

でZMUSICを常駐させ，

A>ZP filename

で即演奏できます。

GS音源を持っている人はリストをちょこっといじって音色番号11：MUSIC BOX（オルゴールの音色）で演奏させるように改造してみてはどうでしょうか。

さて，最近内蔵音源の曲データの投稿が減ってきています。いまOh!X LIVEは内蔵音源が狙い目かもしれないなんていう意味ありげな捨てゼリフを残しつつまた来月。

(Z.N)

リスト1　暁の戦士

```
================  FF5_AKA.ZMS  ===================
  1: .comment FF5 ～ 暁 の 戦 士 ～          Presented By MaSa
  2: (i)
  3:
  4: (m1,2600)(aMIDI1,1)
  5: (m2,1100)(aMIDI2,2)
  6: (m3,1000)(aMIDI3,3)
  7: (m4,1000)(aMIDI4,4)
  8: (m5,1000)(aMIDI5,5)
  9: (m6,1000)(aMIDI6,6)
 10: (m7,1000)(aMIDI7,7)
 11: (m10,1000)(aMIDI10,10)
 12: (m11,1000)(aMIDI10,11)
 13:
 14: / Exclusive ------------------------------
 15: .roland_exclusive $10,$42={$40,$00,$7F,$00}
 16:
 17: .sc55_reverb ={4,3,0,70,30,30,0}
 18: .sc55_chorus ={3,0,60,10,70,8,5,0}
 19: .sc55_v_reserve ={2,2,2,2,2,2,2,2,2,4,0,0,0,0,0,0}
 20:
 21: /----------------------------------------
 22: (o132)
 23: (t1)        @i$41,$10,$42
 24: (t2)        @i$41,$10,$42
 25: (t3)        @i$41,$10,$42
 26: (t4)        @i$41,$10,$42
 27: (t5)        @i$41,$10,$42
 28: (t6)        @i$41,$10,$42
 29: (t7)        @i$41,$10,$42
 30: (t10)       @i$41,$10,$42
 31: (t11)       @i$41,$10,$42
 32:
 33:
 34: /-----------------------------------------
 35: /Trumpet
 36: (t1) @58 @v123 @u125 @p61 @q16 o4 @k-1
 37: (t1) @e60,70 @h36 @s3 @m36 r1
 38: (t1) r1 r1
 39: (t1) [do]18
 40: (t1) g4g4q4fa+@q16g4q4   gd+f<cq8>g2@q16
 41: (t1) g4g4q4fa+@q16g4q4   gd+f<cq8>g4q4g12g12g+12q8
 42: (t1) a+4.g+16g16f2   &f2f6g6g+6   @q30 g*384@q16
 43: (t1) g4g4q4fa+@q16g4q4   gd+f<cq8>g2@q16
 44: (t1) g4g4q4fa+@q16g4q4   gd+f<cq8>g4q4g12g12g+12q8
 45: (t1) a+4.g+16g16f2   &f2f6g6g+6   @q26 g*384_3116
 46: (t1) r2^8@q4d+d+@q12g+8.@q4d+q8  g+1
 47: (t1) r2^8@q4ff@q12a+8.@q4fq8     a+1
 48: (t1) r2^8@q4dd@q12g8.@q4dq8      g1@q8
 49: (t1) d+4.@q4dd+@q8f4.@q4d+fq8  g4.fga+4<d4
 50: (t1) r2^8@q4>d+d+@q12g+8.@q4d+q8 g+1
 51: (t1) r2^8@q4ff@q12a+8.@q4fq8     a+1
 52: (t1) r2^8@q4dd@q12g8.@q4dq8    - g1@q12~
 53: (t1) a4.q4aa@q12a4.q4a16a@q8  b4b4q4b8b8b8b8@q16
 54:
 55:
 56:
 57: (t2) @58 @v119 @u125 @q16 o4 @p67 @k1
 58: (t2) @e60,70 @h36 @s3 @m36 r1
 59: (t2) r1r1
 60: (t2) [do]18
 61: (t2) c4c4q4d+dc4        c>a+<d+dq8c2@q16
 62: (t2) c4c4q4d+dq8c4q4  c>a+<d+dq8c4q4c12c12c12q8
 63: (t2) c+4.c+16d+16c+2&  c+2c+6d+6f6   @q30 e*384 @q16
```

```
 64: (t2) c4c4@q8d+d@q16c4@q8    c>a+<d+d@q33c2@q16
 65: (t2) c4c4@q8d+d@q16c4@q8    c>a+<d+d@q16c4@q5c12c12c12q8
 66: (t2) c+4.c+16d+16c+2&  c+2c+6d+6f6   @q26 e*384_3116
 67: (t2) r2^8cc@q12d+8.q8d16  c1
 68: (t2) r2^8dd@q12f8.q8d+16  d1
 69: (t2) r2^8)a+a+@q12<d8.q8c >a+1
 70: (t2) <c4.>a+<cd4.cd   d+4.dd+g4a+4
 71: (t2) r2^8cc@q12d+8.q8d    c1
 72: (t2) r2^8dd@q12f8.q8d+    d1
 73: (t2) r2^8>a+a+@q12<d8.q8c >a+1@q12<~
 74: (t2) f4.q4ff@q12f4.q4ff@q8  g4g4q4g8g8g8g8@q16
 75:
 76:
 77:
 78: /----------------------------------------
 79: /Strings
 80: (t3) @u111 @p54 o4 @e30,20 @h48 @s3 @m30 r1
 81: (t3) r1r1
 82: (t3) [do]@v100i8@49
 83: (t3) |:c2g2  <~4g4f4d+4c4_>  c2g2  <~g4f4d+4c4_
 84: (t3) c+2f2   g+2<c+2  c1&c1>>:|     <~9i0@49@v98
 85: (t3) |:d+4d8d+8c2&  c2c6d6d+6   f4d+8f8d4&d4&  d1
 86: (t3) d4c8d8>a+4&a+4& a+2a+6<c6d6 | d+1&d+1:|g*384_>
 87:
 88:
 89: (t4) @49 @v98 @u111 o3 @p74 @e30,20 @h48 @s3 @m30 @k1 r1
 90: (t4) r1r1
 91: (t4) [do]
 92: (t4) |:c2g2  <~3g4f4d+4c4_>  c2g2  <~g4f4d+4c4_
 93: (t4) f2a+2  <c+2f2  g1&g1>>:|<<~11
 94: (t4) |:c4>a+8<c8>g+4&g+4&  g+2g+6a+6<c6  d4c8d8>a+2&a+1
 95: (t4) a+4a8a+8g2&  g2g6a6a+6  |<c1&c1:|c1>b1_
 96:
 97:
 98: /----------------------------------------
 99: /Timpani
100: (t5) @48 @v97 @u124 o2 @p70  @e65,45 @k2 14
101: (t5) @y1,$20,$42 r1
102: (t5) c4&@q33c4>g2  q8<c>g<c>gq4
103: (t5) [do]
104: (t5) |:14<c2>g2 q8<c>g<c>a+ 12q4g+d+ g+d+
105: (t5) <c+>g+ a+f <c>g  <c>g:|
106: (t5) |:g+g+ 14q8g+<d+>g+<d+  q4>g+2f2 q8g+<f>g+<f
107: (t5) q4>g2g2 q8g<d>g<d|12q4cc >a+a+:|<q4dd 14q8>ggabq4
108:
109: (t6) @48 @v102 @u124 o3 @p58  @e65,45 @k-1 14 r1
110: (t6) c4&@q33c4>g2  q8<c>g<c>gq4
111: (t6) [do]
112: (t6) |:14<c2>g2 q8<c>g<c>a+ 12q4g+d+ g+d+
113: (t6) <c+>g+ a+f <c>g  <c>g:|
114: (t6) |:g+g+ 14q8g+<d+>g+<d+  q4>g+2f2 q8g+<f>g+<f
115: (t6) q4>g2g2 q8g<d>g<d|12q4cc >a+a+:|<q4dd 14q8>ggabq4
116:
117:
118:
119: /----------------------------------------
120: / Bass
121: (t7) @88 @v78 @u110 @q33 o3 @p64 @k2
122: (t7) @e80,10 12 @y1,$20,$30 r1
123: (t7) c>gq8  <c4>g4<c4>g4q5<
124: (t7) [do]
125: (t7) c>gq8      <c4>g4<c4>a+4q5
126: (t7) |:g+d+:|  <c+>g+a+f   |:3<c>g:|q8  <c4>g4<c4>a+4q5
127: (t7) |:g+d+:|  <c+>g+a+f   |:<c>g:|      g+g+q8
128: (t7) |:g+4<d+4>:|q5   g+f    q8g+4<f4>g+4<f4q5>
129: (t7) ggq8     g4<d4>g4<d4q5
130: (t7) cc>  a+a+
131: (t7) g+g+q8     g+4<d+4>g+4<d+4q5>
132: (t7) g+2f2q8    g+4<f4>g+4<f4q5>
133: (t7) ggq8     g4<d4>g4<d4q5
134: (t7) ddq8>    g4g4a4b4q5<
135:
136:
137:
138: /----------------------------------------
139: / Drums
140: (t10) @49 @v70 @u115 o3 @e70,15 @q1 r1
141: (t10) r1 r1
142: (t10) [do]
143: (t10) a*384 a*384 r1 r1 a*384
144: (t10) a*384 a*384 r1 r1 a*384
145: (t10) a*384 a*384 a*384 a*384
146: (t10) a*384 a*384 a*384 a*384
147:
148:
149:
150: (t11)  @u110 o2 @q1 r1
151: (t11) |: 'cd8''cd16''cd16' 'cd8'
152: (t11) z75,85,100,110'cd32''cd32''cd32''cd32'
153: (t11) z117,-17'cd8''cd8'  'cd16''cd16''cd16''cd16':|
154: (t11) [do]
155: (t11) |:32'c8d''cd16''cd16' 'c8d'
156: (t11) z75,85,100,110'cd32''cd32''cd32''cd32'
157: (t11) z117,-17'c8d''c8d'  'cd16''cd16''cd16''cd16':|
158:
159:
160: (t1) [loop]
161: (t2) [loop]
162: (t3) [loop]
163: (t4) [loop]
164: (t5) [loop]
165: (t6) [loop]
166: (t7) [loop]
167: (t10) [loop]
168: (t11) [loop]
169:
170:
171: (p)
```

リスト2 暁の戦士用カウンタ表示

```
 1:00000240 00001800  2:00000240 00001800  3:00000240 00001800  4:00000240 00001800
 5:00000240 00001800  6:00000240 00001800  7:00000240 00001800 10:00000240 00001800
11:00000240 00001800
```

リスト3 STAR GAZERII

```
===============  STAR_2.ZMS  ==================
  1: .comment - STAR GAZER II -  For SC-55(mkII) by KRC '94/04/0
7
  2:
  3: /    PREPRIMER II より
  4: /  ~  STAR GAZER II  ~
  5: /      from Dragon Slayer 6 ---
  6: /                    船、ゲームオーバー、エンディング
  7:
  8: /    Programmed by  Kurachi Kazuhiro
  9: /                          1994/ 4/ 7 Thu.
 10:
 11: (i)(b1)
 12:
 13: (m1,2000)(a1,1) /この曲が入っているCDは、夏のイメージで作られ
 14: (m2,2000)(a2,2) /たそうなので、夕凪の浜辺で南十字星を見ながら聴
 15: (m3,2000)(a3,3) /くのが「◎」。約五分間たのしめます......
 16:
 17: .sc55_init $10
 18: .sc55_reverb $10={4,4,0,100,64,0,0}
 19: .sc55_chorus $10={6,0,94,14,60,13,00}
 20: .sc55_v_reserve $10={0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0}
 21:
 22: /  Left Hand
 23: / I
 24: (t1) @i$41,$10,$42 i0@1@p50@k0q818 @e110,64 r*432
 25: / A
 26: (t1) @d0o3@v127
 27: (t1) t66@u55 'g<d'210 'f<c'210 @u40'g<d'240
 28: (t1) t70@u60 'e-1b-' 'd-1a-' @u45'da'264
 29: / B
 30: (t1) o2@d1
 31: (t1) |:t72@u45g<dt-5@u+5a@u-10t+5b2 cg<t-5@u+5dt+5e-*140:|>
 32: (t1) |:t72@u50e-b-<t-5@u+5fg^2:|>r8|:t72@u55e-b-<t-5@ufg^2:
|>
 33: (t1) t72@u50da<t-8@uef^2>> t72ug<@udt-8@u-15a@u+10b*228
 34: / C
 35: (t1) t66@u45g<d@u+5f+2> t72f+4@u-10e@u+5b<t-8@u-10f+t-8g2>>
 36: (t1) t72@u55b<f+t-8u+5<c+t-8d4.>>t72r4@u45g<u+5dt-8u+5at-8b
2^8
 37: (t1) t72@u55g<@u-5dt-5f+*142> t72@u45eb<t-8@u+5f+t-8g2>>
 38: (t1) t72@u55b<f+t-8u+5<c+t-8d4.>>t72r4@u45g<u+5dt-8@uat-8b2
>
 39: (t1) t72@u45g<@u+5dt-8@u-10at-8b^2
 40: / D
 41: (t1) uda<t-6@u+5e@uf2 >> t72@u55g<@u+5dt-8@u-10at-8b^2
 42: (t1) t72ucg<@u+5t-8d@u-10'ce'96,1 >> t72@u45f<ct-8@u+5gt-8a
^2>
 43: (t1) t72ua<@uet-8@ubt-8<c>>t66@u35f<@u55cf4
 44: (t1) t72@u55c@u-10g<@u55e4>>b<@u-10g+<@u+10d4>>
 45: (t1) @u60a<e<c4>> @u65g+<@u-15e<@u+10c4r16>
 46: / E
 47: (t1) @u35@d0e*256> @d1@u50f+1 @u25g*400 <
 48: (t1) t72@u20g*0&@u45a-*624
 49: / A'
 50: (t1) @d0o3@v127
 51: (t1) t70@u55 'g<d'210 'f<c'210 @u40'g<d'240
 52: (t1) t70@u60 'e-1b-' 'd-1a-' @u45'da'264
 53: / B'
 54: (t1) o2@d1
 55: (t1) t72@u45g<dt-5@u+5a@u-10t+5b2 cg<t-5@u+5de-*140>
 56: (t1) t72@u50e-b-<t-5@u+5fg^2>r8t72@u55e-b-<t-5@ufg^2>
 57: (t1) t72@u50da<t-8@uef^2>> t72ug<@udt-8@u-15a@u+10b*180
 58: / C'
 59: (t1) t66@u45g<d@u+5f+2> t72f+4@u-10e@u+5b<t-8@u-10f+t-8g2>>
 60: (t1) t72@u55b<f+t-8u+5<c+t-8d4.>>t72r4@u45g<u+5dt-8@uat-8b2
^8>
 61: (t1) t72@u50g<@u+5dt-8@u-10at-8b^2
 62: / D'
 63: (t1) uda<t-6@u+5e@uf2^8 >> t72@u55g<@u+5dt-8@u-10at-8b^2
 64: (t1) t72ucg<@u+5t-8d@u-20'ce'96,1 >> t72@u45f<ct-8@u+5gt-8a
^2>
 65: (t1) t72ua<e@ut-8bt-8<c^2>> t72ug<t-6et-6@ubt-6<c^2>>
 66: (t1) t72uf<ct-6@u-10gt-8a-^2 t72uf<ct-6@u+5gt-8a-2.
```

▶なぜか，ハガキが2枚ついていた。話には聞いていたが，自分にそれが当たるとは恐るべし，東京，九〇九電機。来月からは盛岡でX68000ユーザーを10人は増やすぞ。

畠山　太一(20)青森県

```
 67: (t1) t72@u25c*0@u50g2>@u45b*0<@u35a-2>
 68: (t1) t72@u50a*0<@u40e2> t72@u45a-*0<@u50f2>
 69: (t1) t60@u45c@u-10t-8g<@u+10t-8d@u+5t72e*384
 70:
 71: /  Right Hand
 72: / I
 73: (t2) @i$41,$10,$42 i0@1@k0@p50q818 @e118,74 r*432
 74: (t2) @d1o4@v127 r*1
 75: / A
 76: (t2) |:@u55bg<c>@u-5b*138:| uat-6@ub<t-6@uat-12@ub*167>
 77: (t2) |:@u60b-g<c>@u-5b-^2:| ua<t-6@uct-6@ugt-12@ua1
 78: / B
 79: (t2) |:@u50g4.@u+5d2 @u-10{fe-de-}4&e-*164:|
 80: (t2) @u55b-ag2@u-8fffg1 u'fb-''e-a''d2g'@u|:3'cf':|'d2..g'
 81: (t2) @u45'cf'@u+10'dg'168 r*300
 82: / C
 83: (t2) @u55dc+>@u+5b2uaaab2.< @u65dc+>@u-10b2 uaa@ua@u+5b2..<
 84: (t2) @u50dc+>@u+5b2ua@u-8a@u-5ab2.<@u60dc+>@u-10b2 aa@uab2.
 85: (t2) @u55r2^8g<t+8@u+5dt+4et+4
 86: / D
 87: (t2) f4.>@ua4.<c& {c>bab}4&b2.< @u55e4.>@u+5g4.b{baga}4&a2.
 88: (t2) @u60b<@uceg@p112b>@p50@u70g+<@u55cf
 89: (t2) @u65@p90e@p50g<t-8@u+5@p90c@u@p50dt+8@p90e@p50f4@u-5@p
90e>
 90: (t2) ub<cet-4@p50g@u+5b4.<t+4@u-5c8.
 91: / E
 92: (t2) @l64o4@u55'ceg' @u-15'eg<c' @u-5'g<e' @u'<cg'
 93: (t2) l8o5@u45'e1b'
 94: (t2) @l64o4@u30'ceg',1 @u+10'eg<c',0 'gb<e' @u-6'<eg'208,1
 95: (t2) l8o4@d0@u45q3a-@u+5q1b-<@p80ct-10@ud-@p50e-r4@u+20fr2.
 96: (t2) t72o4@u60q1a-@u+5b-<ct-10@ude-t-8r4@ufr2 >q8@d1
 97: / A'
 98: (t2) |:@u55bg<c>@u-5b*138:| uat-6@ub<t-6@uat-12@ub*166>
 99: (t2) |:@u60b-g<c>@u-5b-^2:| ua<t-6@uct-6@ugt-12@ua1
100: / B'
101: (t2) @u50g4.@u+5d2 @u-10{fe-de-}4&e-*164
102: (t2) @u55b-a@ug2..@u-10f@u+5g2.. @u45'cf'@u+10'dg'168 r*252
103: / C'
104: (t2) @u55dc+>@u+5b2 uaaab2.< @u65dc+>@u-10b2 uaa@ua@u+5b2..
105: (t2) @u55r2^8t+10g<@u+5t+6de
106: / D'
107: (t2) @u50f2^8>@ua4<c&{c>bab}4&b2.<@u55e4.>@u+5g4.b&{baga}4&
a2.
108: (t2) @u50b4.<c^2> @u+10b4.<c^2 g4.f^2 @u-10g4.f2.
109: (t2) @d0q2@u40eg<ct-8d @u+10e2>
110: (t2) q3@u50c@p90et-4@p50at-8@u+5@p90b<
111: (t2) q1@ut-8@p50c4q8t-6d4 @u-10e*456
112:
113: /  Seashore
114: (t3) @i$41,$10,$42 i0@123@p90@k0q8@l300o4 @u60 @e127,70 r4.
115: (t3) |:4@v30c_4c¯c>c_c¯>c_c<<c_>c¯¯c_>c¯c:|
116: (t3) @v30c_<<c__>c_c
117:
118: /  Perform
119: (p)
```

リスト4 STAR GAZERII用カウンタ表示

```
1:000036F6 00000000   2:000036F6 00000000   3:00003D38 00000000
```

リスト5 SAY ANYTHING


```
===============  Say_A.zms  ==================
  1: (i)(b1)
  2:
  3: .comment    ~  Say Anything ~
  4:
  5: .comment              ( X )
  6:
  7: .comment          作詞/作曲  YOSHIKI
  8:
  9: .comment          プログラム  塚本岳彦
 10:
 11: .comment              MIDI-module SC-55
 12: .comment            for  Z-MUSIC system
 13:
 14: /------  Initialize MIDI-module  ----------
 15:
 16: .Roland_exclusive  $10,$42 ={$40,00,$7F,00}
 17: .sc55_print $10 "  Say Anything      ***> X <***"
 18: .sc55_reverb $10={$03,$04,$00,$60,$70,$30,$00}
 19: .sc55_chorus $10={$03,$00,$40,$28,$50,$08,$13,$00}
 20: .sc55_v_reserve $10={2,2,3,1,4,3,3,1,2,3,0,0,0,0,0,0}
 21:
 22: /-----   Track Assign   -----
 23:
 24: (m1,5000)(aMidi1,1)  /Vocal1
 25: (m2,3000)(aMidi2,2)  /Vocal2
 26: (m3,3000)(aMidi3,3)  /D.Guitar1
 27: (m4,3000)(aMidi4,4)  /Strings1
 28: (m5,5000)(aMidi5,5)  /Strings2
 29: (m6,6000)(aMidi6,6)  /Piano1
 30: (m7,3000)(aMidi7,7)  /Piano2
 31: (m8,3000)(aMidi8,8)  /Bass
 32: (m9,5000)(aMidi9,9)  /Vocal1 echo
 33:
 34: (m10,3000)(aMidi10,10) /Drums1
 35: (m11,3000)(aMidi10,11) /Drums2
 36: (m12,3000)(aMidi10,12) /Drums3
 37:
 38: /-----   Data setting   -----
 39:
 40: (t1) @i$41,$10,$42@e94,127
 41: (t2) @i$41,$10,$42@e44,84
 42: (t3) @i$41,$10,$42@e94,64
 43: (t4) @i$41,$10,$42@e108,12
 44: (t5) @i$41,$10,$42@e54,88
 45: (t6) @i$41,$10,$42@e117,20
 46: (t7) @i$41,$10,$42@e117,20
 47: (t8) @i$41,$10,$42@e34,94
 48: (t9) @i$41,$10,$42@e94,127
 49: (t10) @i$41,$10,$42@e127,80
 50:
 51: /-----     MML DATA     -----
 52:
 53: /Vocal
 54:
 55: /[Setting]
 56: (t1) t75 v12_4 o4 q8 l16 @20 @k1 @s5 @h5 @m30 @u95 r1 q8
 57: (t1) [K.Sign #f]
 58:
 59: /[Intro]
 60: (t1) |:17r1:|r4
 61: /A
 62: (t1) u+32|:q7r8gggg8q8|f8ed8r8u-16q7ddu+16g8g8q8a8gg&g4r4:|
 63: (t1) gf8eu-16d8ud8q6e d4r2.q7
 64: (t1) u+32r8ggg8.gq8f8ed8.u-16q7dd u+16g8g8q8a8gg&g4r4q7
 65: (t1) r8gggg8ggg8.eg8g _16|:15¯2a&:|¯a_16uq8
 66: /B
 67: (t1) |:8r1:|
 68: /C
 69: (t1) q7u+32r8ggg8q6gq8gf8eq6d8.u-16q7dd u+16g8g8q8a8gg&g4r4
 70: (t1) q7u-8 r8gggggf8ed8u-8ddq8ue& ed4..r2
 71: (t1) q7u+24r8gggg8q8gf8ed8.u-8q7dd u+8g8g8a8u-8gg&g4r4
 72: (t1)   u+16r8gggg8gg8g8q8fg8a& q7a4&|:3¯2a&:|¯aq8a4b4
 73: /D
 74: (t1) |:¯8[$]q7ag8g8.u-16ddu+16aaaaaa8b u-16q8a8.g&g4r4_u+16
b4
 75: (t1)   ¯     q7ag8g8.u-16eeu+16aaaaaa8[*]b q8a4r4_a4b4
 76: (t1)   ¯     q7ag8g8.u-16d8u+16a8a8a8ab u-16q8a8.g&g4r4_u+16
b4
 77: (t1)   ¯     q7ag8g8 u-16eeeu+16aaaaaa8q8b u-16a2r2_16u|
 78: /E
 79: (t1) u+16q6r8e8q7eg8f8.e8d8>u-16aau+16 b8<dq8d&d4r2
 80: (t1) u+16q7r8eeeg8f8.f8u+8a8a8 u+8q8{bag&}4g4r2
 81: (t1) u-16q7r8eeeg8f8.e8d8>a8  b8<u-16dq8d&d4r2
 82: (t1) u+16q7r8eeee8e8.e8de8. u+16r8gggg8g8.g8eg8¯16a& a1
 83: (t1) r2_8a4b4
 84: /F(D')
 85: (t1) ¯q7ag8g8.u-16q6d8u+16q7aa8a8a8b a8gq8g&g4r4_b4
 86: (t1) ¯q7ag8g8.u-16e8u+16a8a8aa8b& ba8.r4q8_a4b4
 87: (t1) ¯q7ag8g8.u-16d8u+16a8a8aa8b& ba8g&g4r4_b4
 88: (t1) ¯  ag8g8u-16ee8u+16a8a8aaa ¯q8b8a2&_a&_a&_au
 89: / u ¯32 @67 p3 @e44,84 @s5 @h10 @m-20
 90: (t1) u+32 _20 @31 @p95 @e94,64 @m0 @k0
 91: (t1) r4
 92: (t1) r2..cc# d4.def4.fg
 93: (t1) @30p3@m10r4(a4>f)
 94: (t1)      {'df''eg''fa''gb''a<c''b<d'}4<
 95: (t1) {'ce''df''eg''fa''gb''a<c'}4
 96: /G
 97: (t1) [k.sign -b,-e]
 98: (t1) 'b4<d'
 99: (t1) ¯8  (c8,d)&c8f4@b0,1365b32&@b1365@m35b8.&@b1365,0b32@b
0
100: (t1) b2&@m10b8{a@b683a@b0a}8gfedl8
101: (t1) e'>eg''>g<c''ce''eg''ce'@b0,1365'a32<c'&@b1365@m35'a.<
c'&
102: (t1) @b1365,0'a32<c'@b0l16
103: (t1) @b0'a4.<c'@m10{'gb'@b1365'gb'@b0'gb'}8
104: (t1) 'f8.a''g8.b''a<c''fb''a<c'@m35'b4<d'&@b0,-8192@m10'b<d
'@b0'>b<d'
105: (t1) '>a<c''>b8<d' 'd8f''f8a''a8<c'
106: (t1) @b1365'gb'&@b1365,0'g32b'&@b0'g8..b'&r'gb''a<c'
107: (t1) 'b8<d''a<c'|:'gb''fa':|'df'
108: (t1) 'df''>b8.<e''e8g''d8f''e8g''f8a''g8b''a8<c'
109: (t1) @b0,1365'a32<c'&@b1365@m35'a8.<c'&@b1365,0'a32<c'@m10
110: (t1) _8@b0@p95x$40,$11,$35,$60r4
111: (t1) {'f>f''eg''f>f''eg''df''cg'}4{'da''eg''fa''gb''a<c''b<
d'}4
112: /H
113: (t1) [k.sign -b,-a,-e]¯8
114: (t1) <@m50'c4.e'@m10'ce''>b<d'@b-1365,0'df'&@b0@m50'df'&'d4
f'@m10'ce''>b<d'
115: (t1) @b-1365,0'a<c'&@b0@m50'a4.<c'&@b0,-1365'a<c'@b0'g4b'@m
10{gab}4
116: (t1) @m50'c4.e'@m10'fa''eg'@m50'd4.f'@m10'ce''>b<d'
117: (t1) @b-1365,0'd32f'&@b0@m50'd8..f'&'d8.f'&@b0,-1365'df'@m0
@b0
118: (t1) >{'ce''df''eg''df''eg''fa'}4
119: (t1)  {'eg''fa''gb''a<c''b<d''<ce'}4_8
120: /I
121: (t1) [k.sign -a,-b,-d,-e]¯8@m10
122: (t1) '<d4f''fa''a<d'<'df''fa'@m50'g4b'@m10'eg''fa'
123: (t1) @b-1365,0'a32<c'&@b0'a32<c'&@b0,-1365'a<c'@b0
124: (t1) 'e4.a''fa''gb''a8.<c''a<c'@b0,683'a<c'@b0'a<c''gb''a<c
```

▶SHIFT・BREAKにこっそり値上げが書かれている。自分は1万円までなら買ってもいいと思っている(本当か?)。でも，その前に「Oh!電脳」が電撃創刊されたりして……。

角谷　光憲(21)愛知県

```
'&
125: (t1) @m30@d1'a1<d'r1¥40r1¥0@d0>_8
126: (t1) x$40,$11,$35,$40 r1
127: /J
128: (t1) o4v12_4 u @20 p3 @e94,127 @s5 @h5 @m30 @k1
129: (t1) [k.sign b-]
130: (t1) q7rfffffffe8du-16c8.cc uf8f8g8fq8f&f4r4
131: (t1) q7r8ffffffe8dc8u-16ccq8d8 c8.r2.u
132: (t1) q7rffffffffeedc8.u-16cc u+16f8f8g8fq8f&f4r4
133: (t1) q7u+16~8r8fffffff8f8df8u+16q8g& g1
134: (t1) r2a4b!4
135: /K
136: (t1) [k.sign +f]
137: (t1) :|
138: /L(E')
139: (t1) u+16q7r8e8eg8f8.e8d8>a8  b8<udq8d&d4r2
140: (t1) u+16q7r8e8eg8f8.u+16a8a8a8  q8~8ba8g&_g4r2
141: (t1) u-16q7r8eeeg8f8.e8d8>ua8  b8<dq8d&d4r2
142: (t1) u+16q7r8eeee8e8.e8de8. u+16r8gggg8g8.g8eg8~a& q8a1
143: (t1) r2a4b4
144: /M(D')
145: (t1) ~8q7a8gg8.u-16q6d8q7u+16a8q6a8q7aa8b a8gq8g&g4r4b8.a
146: (t1)   q7aggg8.u-16e8u+16a8aaaa8b q8a4r4_a4b4~[d.s.]
147: (t1) [coda]b8q8a4.._b4a4~
148: (t1)                |:q7ag8g&g4r4|q8b4:|r8u-16e8u+16
149: (t1) <c4cd8c&c4r8>aa< dd8d8d8e8d8.|:q8d8.cq7
150: (t1) c>b8q8b&|b4r4<:|>b2. r1r1
151:
152:
153:
154: /Vocal2
155:
156: /[Setting]
157: (t2) v14 o4 q8 l16 @20 @s7 @h15 @m40 @u75 r1
158: (t2) [K.Sign #f]
159:
160: /[Intro]
161: (t2) |:17r1:|r4
162: /AB
163: (t2) |:16r1:|
164: /C
165: (t2) |:7r1:| r2f4f4
166: /D
167: (t2) |:~8dd8d8.>bb<fffff8g8g8e&e4r4g4fe8e8.cceeeeee8gf4r4f
4f4
168: (t2)     dd8d8.>b8<f8f8f8fgf8.e&e4r4g4fe8e8ccceeeeee8gf2r2_
|
169: /E
170: (t2) |:9r1:| r2f4f4
171: /F(D')
172: (t2) ~8dd8d8.>b8<ff8f8f8g f8ee&e4r4g4 fe8e8.c8e8e8ed8g& gf8
.r4f4f4
173: (t2)   dd8d8.>b8<f8f8ff8g& gf8e&e4r4g4 fe8e8cc8e8e8eee g8f4
..r2
174: (t2) _r1r1
175: / u ~32 @66 p3 u @e44,84 q8 @s7@h15@m40
176: (t2) u+52 _40 @31 @p95 @e94,64 @m0 q7
177: (t2) >r4(a4>f)< q5
178: (t2)      {'df''eg''fa''gb''a<c''b<d'}4<
179: (t2) {'ce''df''eg''fa''gb''a<c'}4
180: /G
181: (t2) [k.sign -b,-e]q8
182: (t2) ~16'b2<d'_r2|:6r1:|r2q6
183: (t2) {'ff''eg''ff''eg''df''cg'}4{'da''eg''fa''gb''a<c''b<d'
}4
184: (t2) q8
185: /H
186: (t2) r1r1r1.~8'<c4.e-'_r8
187: /I
188: (t2) [k.sign -a,-b,-d,-e]~8
189: (t2) r1...>'a<d''g<c''a8<d'<'df''ce!''d8f''fa''e!g''f8a'
190: (t2)        'a<d''g<c''a8<d'<'df''ce!''d8f''fa''e!g''f8a'
191: (t2)        'a<d''g<c'
192: (t2) 'a<d''fb''g<c''a<d''g<c''fb''e!a''df'
193: (t2) 'e8g''>b<e''df''e8g'|:'eg''fa''gb'&:|@b0,1365'gb'
194: (t2) @b0'gb''fa''df' 'e!g''ce!''d!f''e!g''fa''e!g''fa''gb'
195: (t2) @b-1365,0'a!32<c'&@b0'a!8..<c'&'a!8.<c'&
196: (t2) @b0,-1365'a!<c'_8>@b0
197: /J
198: (t2) u ~40 @20 o4 p3 @e44,84 q8 @s7@h15@m40
199: (t2) |:8r1:|r2f#4f#4
200: /K
201: (t2) [k.sign +f]
202: (t2) :|
203: /L
204: (t2) |:6r1:|
205: (t2) r8cccc8c8.c8>a<c8. r8eeee8e8.e8ce8f& f1
206: (t2) r2'f4<c''f4<d'
207: /M
208: (t2) ~8'd8<c''db''d8.b''>b8<g'|:'f8<c':|'f<c''f8<c''g<d'
209: (t2)   'f8<c''eb''eb'&'e4b'r4'g8.<d''f<c'
210: (t2)   'f<c''eb''eb''e8.b''c8g''e8<c'|:3'e<c':|'e8<c''g<d'
211: (t2)   'f4<c'r4'f4<c''f4<d'
212: (t2)   'd<c''d8b''d8.b'|:'>b<g':||:5'f<c':|'f8<c''g<d'
213: (t2)   'f8.<c''eb'&'e4b'r4'g4<d'
214: (t2)   'f<c''e8b''e8.b''cg''cg'|:5'e<c':|'e8<c''g8<d'
215: (t2)   'f4..<c''g4<d''f4<c' |:'d<c''d8b''db'&'d4b'r4|'g4<d'
:|
216: (t2)   r8c8 g4ga8a&a4r8e8 aa8a8a8b8a8.|:b8.a ag8g&|g4r4:|g2
.
217: (t2)   r1r1_8
218:
219:
220:
221: /D.Guitar1
222:
223: /[Setting]
224: (t3) x$40,$13,$36,$60
225: (t3) v6 o4 q8 l16 @28 @u127 r1 @p111 _4
226: (t3) [K.Sign #f]
227:
228: /[Intro]
229: (t3) |:17r1:|r4
230: /AB
231: (t3) |:16r1:|
232: /CD
233: (t3) l8
234: (t3) |:3>gb<dg>fa<df >egbg<e>bge ceg<c>>a<eg<c >da<d>a<ed>a
d<:|
235: (t3)     >gb<dg>fa<df >egbg<e>bge ceg<c#>>a<eg<c# >da<d>a<ed
>ad<
236: /E
237: (t3) >|:ceg<c>da<df |>gb<d>b<gd>bg:|egbg<e>bge
238: (t3)    ceg<c>da<df  >gb<d>b<gd>bg
239: (t3) |:cege<c>gec:| |:da<d>a<fd>ad:|<
240: /F l8 @p127 @28 _32 q8
241: (t3) l16 @p48 @31 ~40 @m10
242: (t3) |:>|:>~8g2f2 e1 <_8c2|>~8a2 <_8d1 :|c#2|
243: (t3) ~20d4.def4.fg _8a4.ab<c4._8cc#
244: (t3) d4.def4.fg a4.ab<c2>_16
245: /G(t1 G とほぼ同じ)
246: (t3) [k.sign -b,-e]p3@m0
247: (t3) ~8r4(c8,d)&c8f4@b0,1365b32&@b1365@m20b8.&@b1365,0b32@b
0
248: (t3) b2&@m0b8{a@b683a@b0a}8gfedl8q7
249: (t3) e'>eg''>g<c''ce''eg''ce'q8@b0,1365'a32<c'&@b1365@m20'a
.<c'&
250: (t3) @b1365,0'a32<c'l16
251: (t3) @b0'a4.<c'@m0{'gb'@b1365'gb'@b0'gb'}8
252: (t3) 'f8.a''g8.b''a<c''fb''a<c'@m20'b4<d'&@b0,-8192@m0'b<d'
@b0'>b<d'
253: (t3) '>a<c'q7'>b8<d''d8f''f8a''a8<c'
254: (t3) @b1365'gb'&q8@b1365,0'g32b'&@b0'g8..b'&r'gb''a<c'
255: (t3) 'b8<d''a<c'|:'gb''fa':|'df'
256: (t3) 'df''>b8.<e''e8g''d8f'q7'e8g''f8a''g8b''a8<c'q8
257: (t3) @b0,1365'a32<c'&@b1365@m20'a8.<c'@b1365,0'a32<c'@b0'a2
<c'&r8
258: (t3) @p95@d0@m0r8_8
259: /H(t1 H とほぼ同じ)
260: (t3) [k.sign -b,-a,-e]~8
261: (t3) <@m35'c4.e'@m0'ce''>b<d'@b-1365,0'df'&@b0@m35'df'&'d4f
'@m0'ce''>b<d'
262: (t3) @b-1365,0'a<c'&@b0@m35'a4.<c'&@b0,-1365'a<c'@b0'g4b'@m
0q6{gab}4q8
263: (t3) @m35'c4.e'@m0'fa''eg'@m35'd4.f'@m0'ce''>b<d'
264: (t3) @b-1365,0'd32f'&@b0@m35'd8..f'&'d8.f'&@b0,-1365'df'@m0
@b0q6
265: (t3) >{'ce''df''eg''df''eg''fa'}4
266: (t3)  {'eg''fa''gb''a<c''b<d''<ce'}4q8_8
267: /I(t1 I とほぼ同じ)
268: (t3) [k.sign -a,-b,-d,-e]~8
269: (t3) '<d4f''fa''a<d'<'df''fa'@m35'g4b'@m0'eg''fa'
270: (t3) @b-1365,0'a32<c'&@b0'a32<c'&@b0,-1365'a<c'@b0
271: (t3) 'e4.a''fa''gb''a8.<c''a<c'@b0,683'a<c'@b0'a<c''gb''a<c
'&
272: (t3) @d1'a1<d'r1r1@d0@e127,20@32r1_8>>
273: /J
274: (t3) [k.sign b-]
275: (t3) @m20 _8¥20'f1a'¥0v11@b0@31
276: (t3) @p111 @m10 @e94,64 q8
277: (t3) |:6r1:| ~16c4.cde4.ef f#2 _32d2@p49
278: /K
279: (t3) [k.sign +f]
280: (t3) @31v11~4@p49
281: (t3) _8:|d1
282: /L
283: (t3) |:c2d2 >~g1 |<_c2d2 e2>~e2<_:| _16|:'>g<c8g<c'|:7'>g<c
8g':|:|
284: (t3) '>a<d4.a<d''>a<d8a<d''>a<d2a<d''>a<d1a<d'~8
285: /MN(F')
286: (t3) |:|:4 |:>~g2f2 e1 <_c2|>~a2 <_d1 :|c#2
287: (t3) d1:|¥6:|
288:
289:
290:
291: /Strings1
292: /[Setting]
293: (t4) x$40,$14,$30,$60,$30,$40,$40,$45,$30
294: (t4) v10 o5 q8 l16 @49 @s1 @h20 @m10 @u127 r1
295: (t4) [K.Sign #f]
296:
297: /[Intro]
298: (t4)   |:c8.>b8&b2&|dgb<:|>bb8<e2&e8f8g8>b<c&c4&ce8~8d&~8d2
&_16
299: (t4) r<|:c8.>b8&b2&|dgb<:|>bb8<c2&c8>b8a8ba&|:6_8a8&:|_8a8r
~16
300: (t4) <r8|:c8.>b8&b2&|dgb<:|>bb8<e2&e8f8g8>b<c&c4&c~24e8d&~2
4d2&
301: (t4) r_8|:c8.>b8&b2&|dgb<:|>bb8<c2&c8>b8a8ba&a1& ~16a2<~8d2
&r_24@d0>
302: /AB
303: (t4) |:16r1:|
304: /C
305: (t4) |:8r1:|
306: /D
307: (t4) _16>b2<d2 e1 g1 a1 b2<_16d2 e2..f8 g2a4g4 f1>~32
308: /E
309: (t4) _16e2r4d4 >b2<<c4>b4 e2f2 _r4<g4f4e4>~ g2a2
310: (t4) b2<c4>b4 ~>>g4<c4e4_g4< c1 d2.a4f4d4_<f2>~32
311: /F
312: (t4) _32<g2f4d4 e1 e1 d2>a4b4 <d2a4f4
313: (t4) e2..f8 g2a4g4 f1& f1 r1. >>~32{fgab<cd}4{efgab<c}4>
314: /G
315: (t4) [k.sign]
316: (t4) _32<d2f2 g2d4f4 e-1 r1 f1 d1 g4f4e-4>b-4 <f1>
317: /H
318: (t4) ~8<e-2f2 a-1 e-2f2 e-1>
319: /I
320: (t4) ~16a-2b-2<c2e-2d-1&d-1e-1e1>~8
321: /J
```

▶まさか，フロッピーディスクがないために困るなんて考えたこともありませんでした。2Ｄのフロッピーディスクがなくなったときと同じく歴史は繰り返されるものなのです。でも，8インチフロッピーは作ってるんですよ。　浅野　憲(24)東京都

```
322: (t4) _40@50r1r1r1.>f8.e&e4 c1 d1d1@49¯24r1.<a4b4¯16
323: /K                                     -
324: (t4) _16<g2f#2 >b1<r2g2 >a2a4b4 b2a4f#4 g2.<f#4 g2a4g4 f#1>
-
325: /L
326: (t4) _16r1.<c4>b4 r1 r1 b2<d2 >b1 ¯c1&c1 d1&d1
327: /M
328: (t4) _16<g2f#4d4 e1>
329: (t4) <e2g2>a2a4b4 d2.f#4 <e1 g2a4g4 f#1 g2f#4d4 e2g2
330: (t4) c2e2 d4.e8f#2 g2a2 a8g8g4f#4¯8e8d8 ¯8e2a4g4 f#1
331: /N
332: (t4) |:g2a2. g2>g4 <e2>g2a2&a8f#8g8a8 b2<d4a4 g2.f#4 e2g2a1
333: (t4) >g2a4.d8 b2..<e8d8.>b8.e8b8a4g8 d2a4<d4
334: (t4) >b2<d2 e2&e8>f#8g8a8 a4b4<e4.f#8g4f#4e4d4¥6 g2f#4d4e2>
b2
335: (t4) r2<e4d4 d2e4f#4 g2a4d4 a8g8g4f#4e8d8 e2a4g4f#1:|
336:
337:
338:
339: /Strings2
340: /[Setting]
341: (t5) v8 o4 q8 l16 @50 @u111 r1
342: (t5) [K.Sign #f]
343:
344: /[Intro]
345: (t5) '>g1b<d' '>f1b<d'&'>e1b<eg''c2eg'¯8'd4g'&
346: (t5) ¯16a4&r_24
347: (t5) '>g1b<d<d'&'>f1b<d#<d#' 'c1eg<g' 'd2fa<d''f4a'&
348: (t5) ¥85_16'a4<c'&r¥0v4
349: (t5) 'g1b<d' '>b8.<f<d'&'>b2.<f<d'&r 'e4g''f4a''g2b'
350: (t5) |:¯4'e4g<c'&:||:¯4'f4a<d'&:|r¯8
351: (t5) '>g1b<d<g'&'>f1b<d<f' '>e1<cg<g'
352: (t5) '>a2<d<d'&'c2e<d'&¯8'd2f<d'&¯8'f2a<f'&'fa<f'_8@d0
353: /A
354: (t5) |:8r1:|
355: /B
356: (t5) |:5r1:| r2_8>e4<'>f4<d#'¯8 e4d4c4>g4 f2g4¯8f4<_
357: /C
358: (t5) >b2a2 b1 <e1 >a2<¯8c4¯f4_16 >b2a2 b2<g2 e1 |:4f4&¯4:|_
16
359: /D
360: (t5) |:8r1:|
361: /EF
362: (t5) |:21r1:|
363: /G
364: (t5) [k.sign]
365: (t5) |:'>b-2<d''>f2a<c' '>g1b-<d'|'>a1<ce-' '>a1<cf':|
366: (t5) '>g1b-<e-' '>a1<cf'
367: /H
368: (t5) |:'>a-2<ce-''>b-2<df' | '>b-1<e-g':| 'c1e-g'
369: /I
370: (t5) '>a-2<d-f''>b-2<e-g''c1e-a-''>a-1<d-f'&r1'>b-1<e-g''c1
e!g'
371: /J
372: (t5) _16r1r1r1.c2 >a2g2 ¯8'f2>d''f4d''a4c' 'b-2>b-'&'b-2>g'
373: (t5) ¯@120r1rr64d2@50r4.r32.<
374: /K
375: (t5) |:8r1:|
376: /L
377: (t5) |:'>g2<ce''>a2<df#'|'>g2b<d''>g4<cd''>g4b<d':|'>b1<eg'
378: (t5)   '>g2<ce''>a2<df#' '>g2b<d''>g4<cd''>g4b<d'
379: (t5) |:'>g1<ce'&:|'>a1<df#'&'>a1<df#'
380:
381:
382:
383: /Piano1
384: /[Setting]
385: (t6) v14 o4 q8 l16 @2 @u95 r1
386: (t6) [K.Sign #f]
387:
388: /[Intro]
389: (t6) @49_16u+16r1>f2a4g8f8r
390: (t6) |:11r1:|_r8r2<a2r2c4>b4<r1r1¯32u@2r
391: /A
392: (t6) |:|:'>b4<dg':||:'>b4<df':| |:3'>b4<eg':| 'e8g''d8f'
393: (t6) |:4'c4eg':||:'>a4<df':|u+16'>a4<dg''>a4<df'u
394: (t6) |:'d4gb':||:'d4fa':| |:'>b4<eg':|'g4b<e''g8b''f8a'
395: (t6) |:|:'c4eg':|u-16:|u |u+16'>a8<df'u-32>a8<d8f8g8f8g8a8u
396: /B
397: (t6) |:u+16'd8.g<c'u'dgb'&'d4gb'u-16|d8g8bdgb
398: (t6)      u+32'd#8.f<c'u'd#fb'&'d#4fb'u-16b8f8d#8b8
399: (t6)      u+32'g4b<e'u'g4.b<e'@1u+16'a8<f',2<g8>b<c&
400: (t6) c4&c>@2u+16'g8<ce',0'fa<d'&'f4a<d'uu-16du+8eufa
401: (t6) :|g4rdgb
402: (t6) u+32'd#8.f<c'u'd#8.fb'u-16d#8b8f8d#8b8
403: (t6) u'e4g<c''e4.gc''e8b'a8ba& a4 u+8|:'>a4<df'|'>a4<dg':|u
404: /C
405: (t6) :|u-8|:|:'>a4<df':|u+16:|u
406: /D
407: (t6) |:'>b4<dg':||:'>a4<df':|
408: (t6) '>b4<eg''>b8.<eg'u+8'gb<e''g8.b<e''egb''e4gb'
409: (t6) 'c4eg'u|:3'c4eg':|
410: (t6) |:|:'>a4<df':|u+8:|u |:'>b4<dg':||:'>a4<df':|
411: (t6) |:'>b4<eg':|'g4b<e''e4gb'
412: (t6) |:'c4eg':|u-8|:'c#4ea':| u-8|:'>a4<df'u:|'>a4<dg''>a4<
df'
413: /E
414: (t6) |:'>g4<ce':||:'>a4<df':| '>b8<dg'd8g8b8u+16'd4g<c'u'd4
gb'
415: (t6) u+16|:'>g4<ce'u:||:'>a4<df':|
416: (t6) '>b8.<eg'u+8'egb'u+8'e8.gb'u-8'g4b<e'|:'b<eg'u:|
417: (t6) 'a8<df'u+8'gb<e'
418: (t6) u+16'e4g<c'u'>g4<ce'|:'>a4<df':|
419: (t6) |:'>b4<dg':|'d4g<c''d4gb' u+16|:|:4'>g4<ce':|u+8:|uu+8
420: (t6) f8>a8<d8>a8<f8d8>a4< u+8'>a8<df'u+8>fa<dfa<d>u|:'a4<df
':|
421: /F
422: (t6) u+24'b4<dg'u'>b4<dg'|:'>a4<df':|
423: (t6) '>b4<eg'u-8|:'>b8.<eg''>b<eg':|'>b4<eg'u
424: (t6) |:4'c4eg':| |:4'>a4<df':| u+8|:'>b4<dg'u:||:'>a4<df':|
425: (t6) '>b4<eg''>b8.<eg'u+16'egb''e8.gb'u'gb<e''g4b<e'
426: (t6) |:'c4eg'u+8:|u|:'c#4ea'u+8:|u|:16'f4a<d':|
427: /G
428: (t6) [k.sign -b,-e]_8
429: (t6) |:'e4gb':||:|:'c4fa':| |:'d4gb':|u+16'b4<dg'u'g4b<d'|
430: (t6) |:4'c4eg':||:4'c4fa':| |:'d4fb':|:||:4'e4gb':||:4'f4a<
c':|
431: /H
432: (t6) |:|:'c4ea-':||:'d4fb':| | |:4'e4gb':|:| |:4'c4eg':|¯8
433: /I
434: (t6) [k.sign -b,-a,-d,-e]
435: (t6) |:'d4fa':||:'e4gb':||:4'e4a<c':|
436: (t6) |:8'd4fa':||:4'e4gb':||:'e!4g<c':|'e!2g<c'
437: /J
438: (t6) [k.sign -b]_24
439: (t6) <u'f8a'c8fcfau+8'c8.eg'uf8.e8
440: (t6)  u+16'>a4<df'u>fa<deu+24>'a8.<f'u-8'a8.<e'u-8'a8<d'
441: (t6) u+8'f4b<d'u>b<dfau+16'd4fb'u-8'f4b<d'
442: (t6) u+16'e4.g<c'ucu+8'fg<c'u+24'f8.g<c'u-16e8.u-8fg
443: (t6) u+8'c4.fa'uabu+16'e8.g<c''<c8.f'u-8'<c8e'
444: (t6) u+8'a4<df'&ru-8a<dfu+8'd8.fa'ug8.f8
445: (t6) ¯16u+8>'f4.b<d'b<d>'d4gb'u+8'<d4fa'¯8
446: (t6) |:'e4.g<c'|:'eg<c':|:| 'f#2a<d'u-8|:'f#4a<d':|u
447: /K(D')
448: (t6) [k.sign +f]
449: (t6) |:'>b4<dg':||:'>a4<df':|
450: (t6) |:'>b4<eg':|u+16e8>b8g8e8<u
451: (t6) |:4'c4eg':|
452: (t6) |:|:'>a4<df':|u+8:|u |:'>b4<dg':||:'>a4<df':|
453: (t6) |:4'>b4<eg':|
454: (t6) |:'c4eg':|u-8|:'c#4ea':| u-8|:'>a4<df'u:|'>a4<dg''>a4<
df'
455: /L(E')
456: (t6) |:'>g4<ce':||:'>a4<df':| |:'>b4<dg':|  u+16'd4g<c'u'd4
gb'
457: (t6) u+16|:'>g4<ce'u:||:'>a4<df':|
458: (t6) '>b8<eg'u+8>gb<efgab8.bu+8<efga
459: (t6) 'e4gb'u>'g4<ce''f4a<d''d4fa' u+16|:'>b4<dg'u:|
460: (t6) u+16'd4g<c'u 'd4gb' u+8|:|:4'>g4<ce':|u+8:|uu+16
461: (t6) '>a4.<df''>a8<df'u+8
462: (t6) <<d.>a32ad32d32>a.f32fd32d32
463: (t6) >@d1{fdab<cd}4@d0@d1{efgab<c}4@d0@d1
464: (t6) {defgab}4@d0@d1{<cdefga}4u
465: /M(F')
466: (t6) b4@d0>>'>b4<dg'|:'>a4<df':|
467: (t6) '>b4<eg'u+8'e8.gb'u+16'gb<e''g8.b<e'u-8'egb''e4gb'u-8
468: (t6) |:4'c4eg'uu+16:| |:4'>a4<df'u:| u+16|:'>b4<dg'u-8:|u+1
6|:'>a4<df'u-8:|
469: (t6) u+16'>b4<eg'u-8'>b8.<eg'u+16'egb''e8.gb'u-8'gb<e''g4b<
e'
470: (t6) u+8|:'c4eg'u-8:|u+16|:'c#4ea'u-8:|u+8|:4'>a4<df'u:|
471: (t6) u+8|:'>b4<dg'u+8:|uu+8|:'>a4<df':|
472: (t6)  u+8|:4'>b4<eg':|u-8|:|:4'c4eg':||:4'>a4<df':|
473: (t6)  u+8|: '>b4<dg':|u|:'>a4<dg'u+16:|
474: (t6) uu+8|:4'>b4<eg':|    |: 'c4eg':||:'c#4ea':|
475: (t6)       |: '>a4<dg'u:||u+8'd4fb''d4fa'u
476: /N
477: (t6) [$]
478: (t6) u+16|:'>b4<dg'u-8:|u+8'>a8<df'>a8<<d8f8>
479: (t6)       |:4'>b4<eg'u:|u+8:|uu+8'>a4<dg''>a4<df'
480: (t6)    |:|:'>b4<dg':||:'>a4<df'~:| |'>b4<eg'
481:
482: (t6) u+8 <gfe>bgfe>b<fe>ba f8e8<c>b<ceu-8'c8.eg''c8eg'a8.
483: (t6) 'd4fb''d4fa''>a4<dg''>a4<df' u+8d8bbbaaaa<ccc8>b8b8
484: (t6) b8bb<eeeeffffgggg |:aaaagggg|bbbb:| ffffeeeeddd ¥6
485: (t6) >bbbbaaaa<cccc>bbbbbbbb<eeeeffffggggaaaabbbb<ccccdddd
486: (t6) {ededed}8.effffggggaaa>>u-8:|
487: (t6) |:4'>b4<eg':||:'c4eg':|'c#4ea''e4a<c#'|:'>b4<df':|u+8
488: (t6) 'd4fb''c4fa'u[d.s.]
489:
490:
491: /Piano2
492: /[Setting]
493: (t7) v14 o2 q8 l16 @2 @u95 r1
494: (t7) [K.Sign #f]
495:
496: /[Intro]
497: (t7) |:17r1:|r4
498: /A
499: (t7) |:u+16g2ub2 <e2..   d8    c2> a2 <d4..>u+8au+16d2
500: (t7) u-8 g2ub2 <e2..u+8d8 u+8c2>ua2 u+32d1u|
501: /B
502: (t7) u+32g4u-40<g8b8&b2 >u+40b4u-40<f2. u+24e4.u-8@1>b8<e2@
2u
503: (t7) u-8c8e8g8<c8>u d8fa4&a>
504: (t7) u+32g4u-40<g8b4.u-8bag8u >u+32b8u<f8b2.
505: (t7) u+32c4.u-40>geu+40c2 d4.ua8<d4>u+16d4u
506: /C
507: (t7) :|
508: /D
509: (t7) g2f2 u+32e1 u-16<c2>ua2 d1 g2f2 u+32e1 u-16<c2c#2>u d1
510: /E
511: (t7) <c2d2 g2>g2 <c2d2 u+32e1 u-16c2ud2 g2>g2< |:u+16c1:|
512: (t7) ud4.>d8&d2 u+8d2u+8d4d4u
513: /F(D')
514: (t7) |:g2f2 u+32e1 u-16<c2>ua2 d1 g2f2 u+32e1 u-16<c2c#2>|
515: (t7) d4.def4.fg a4.ab<c4.cc#
516: (t7) d4.def4.fg a4.ab<c2>>u
517: /G
518: (t7) [k.sign]
519: (t7) |:b-2a2 g1 <c1 f1>:|
520: /H
521: (t7) |:a-2b-2 | <e-1>:| <c1>
522: /I
523: (t7) <d-2e-2 >a-1 <d-1d-1e-1e1>
524: /J
525: (t7) _24<<f2u+8e2 d1 u+8>b-2g2 u+8c1 u-16f2e2 d1 ¯16>ub-2g2
526: (t7) ¯8<u+16c4.cde4.ef f#2d4d4>u
527: /K(D')
528: (t7) g2f+2 u+32e1 u-16<c2>ua2 <d1> g2f+2 u+32e1 u-16<c2c#2u
```

▶ 久しぶりに学校へ行くと，すでに試験は終わっていた。うっきぃ！

平　勝久(21)大阪府

```
 d1>
529: /L(E')
530: (t7) <c2d2 >g1   <c2d2 u+32e1 u-16c2ud2 g1 |:u+16c1:|
531: (t7) uu+16d4.>u+8d8&d2 <d2d4d4u>
532: /M(F')
533: (t7) [k.sign +f]:|
534: (t7) <|:d2d4d4>| a2f2 u+16e1u-16
535: (t7) <c2>a2 <d1> u+8g2u-8f2 u+8e1u-8 <c4.>u+8b<cc#2u-8:|>u
536: /N
537: (t7) |:
538: (t7) u+16 |:g2u-8f2 u+8e1|u-8<c2>a2 <d1>u+8:||:uu+8 <c4.>b<
cc#2
539: (t7) d1>|u+8g2u-8f2 u+8e1 u-8<c2>a2 <d1>
540:
541: (t7) u<|:|:'g4b<d':||:'f4a<d':| |:4'g4b<e':|   |
542: (t7)     |:'g4<ce':||:'a4<c#e':||:4'f4a<d':|¥6:|
543: (t7)     |:'e4g<c':||:'e4a<c':| |:4'a4<df':|>
544:
545: (t7) u+16g2u-8f2 u+8e1:|u:|
546:
547:
548: /Bass
549: /[Setting]
550: (t8) v12 o1 q8 l16 @36 @u127 r1
551: (t8) [K.Sign #f]
552:
553: /[Intro]
554: (t8) <@50 g1 b1 e2&e8f8g8b8 <c2>d2&r
555: (t8) g1 b1 <c2c8>g8c4 d2&¥85_16d4r4r¥0v12
556: (t8) r1r1r1r2.d4&r
557: (t8) g1 b1 <c1> d1 ‾8d2<‾8d2&d_16>>
558: /A
559: (t8) |:8r1:|
560: /B
561: (t8) @50<|:4r1:| g1 b1 <c1 d1>>
562: /C
563: (t8) @36g4.aa#r8b4<_8f>‾b <_e4.>‾eer8e4b8<_
564: (t8) c4..c>‾a8a4. <_d2d>‾a<_d4.
565: (t8) g4.<_dc>‾b4.g8   e4.>‾eer8e4<_d8
566: (t8) c4..c>‾a8a4. <_d2d4d8>‾a8
567: /D
568: (t8) g2<_f2 e4..>‾ee8e4b8 <_c2>‾a8a4. <_d2d>‾a<_d4d8
569: (t8) g4.<_d>‾gf2 e4..>‾ff8f4<_d8 c4.>‾b<_cc#8c#4.
570: (t8) d4.(a8,b)<_d>‾ar<_fgg#a8>>‾‾
571: /E
572: (t8) <_8c2d8d4a8 g4..>‾gg4b4
573: (t8) <_c4.<_c>‾gd8d4d>‾a <_e4&e<_g8f8e8>‾bgfe8
574: (t8) c2d8d4>‾a8 g2g4b4 <_|:|:7c8:||<_c>‾g:|cc
575: (t8) d4.d2_(a8<d)&a2>‾d4d4>‾
576: /F
577: (t8) g2f2 e1 <_c2>‾a8a4aa <_d2d8d4d8 g4.<_d>‾gf4f8f8
578: (t8) e2e8e4d8 c4.>‾b<_cc#8c4. d4.de>‾f4.fg a4.ab<_c4.cc#
579: (t8) d8ddd8de>‾f8fff8fg a8aaa8ab<_{ccc}4{ccc}4>‾
580: /G
581: (t8) [k.sign -b,-e]
582: (t8) b2a8a4. g2g4b!4 <_c2c8c4gg- f2>‾f4a4
583: (t8) b2a8a4a8 g2<_g4f4 e4..>‾b<_e8e4e8 f2f4g4>‾
584: /H
585: (t8) [k.sign -b,-a,-e]
586: (t8) a4.a8b2 <_e4.e>‾b<_e8e8>‾g4
587: (t8) a4.a8b8b4b8<_c4d4e4f4
588: /I
589: (t8) [k.sign -b,-a,-d,-e]
590: (t8) d2e4>‾b4 a2a4<_c4 |:8d8dd:||:4e8ee:| e!4._(e!8<e!)&>e!
2‾
591: /J
592: (t8) |:7r1:|‾c4.cd!e!4.e!f f#2_d!4d!4>‾
593: /K(D')
594: (t8) [k.sign +f]
595: (t8) g2 f2 e4.. ee8e4b8 <_c2>‾a2 <_d1>‾
596: (t8) g2 f2 <_e4..>‾ee8e4<_d8 c4.>‾b<_cc#8c#4.
597: (t8) d4. (g8<b)&>ggr<_fgg#a8>>‾‾
598: /L
599: (t8) <_8c2d8d4>‾a8 g2g4b4
600: (t8) <_c2d8d4d>‾a <_e4<_efgf8e8>‾bgfe8
601: (t8) c4.<_c>‾gd8d4>‾a8 g2g4b4 <_|:16c8:|
602: (t8) d4.d2_(a8<d)&>‾a2d4d4>‾
603: /M
604: (t8) g2f8f4. e2e8e4b8 <_c2>‾a8a4. <_d2d>‾a<_d4. >‾g2f8f4.
605: (t8) e2<_e4d4 c2c#8c#4&r>‾a<_ d2d4d4 g2f2 e2e8e4>‾b8
606: (t8) <_c2>‾a8a4.<<__d8.>‾a8.d8.a8.d4 >‾g2f2 e2e8e8b4
607: (t8) <_c2c#8c#4c#8 d2d4d4>‾
608: /N
609: (t8) <_|:|:g2f2 e2e8e4d8 | c2>‾a8a4. <_d4.ed8>‾a<_d>‾a<_d4
610: (t8)        g2f2 e2e4>‾b4<_ c2c#8c#4. d2d>‾a<_d>‾a8a<_d8:|
611: (t8) c4.c8>‾a8a4<_e8 d2d8d4d8
612: (t8)        g2f2 e2>‾e8e4<_d8 c4.‾>b<_cc#8c#4c#>‾a
613: (t8) <_d1¥6g2f2 e2e4d4 c2>‾a8a4.<_ d2d>‾a<_d8d4
614: (t8)        g2f2 e2e4>‾b4<_c4.>‾b<_cc#8c#4c#>‾a <_d2d4d4:|
615:
616:
617: /Vocal Echo
618:
619: /[Setting]
620: (t9) v7‾4 o4 q8 l16 @20 @u95 @k-7 r1 r16.
621: (t9) [K.Sign #f]
622:
623: /[Intro]
624: (t9) |:17r1:|r4
625: /A (A,B,C,D,E はトラック1と99%同じ)
626: (t9) u+32|:q7r8gggg8q8|f8ed8r8u-16q7dd u+16g8g8q8a8gg&g4r4:
|
627: (t9) gf8eu-16d8ud8q6e d4r2.q7
628: (t9) u+32r8ggg8.gq8f8ed8.u-16q7dd u+16g8g8q8a8gg&g4r4q7
629: (t9) r8gggg8ggg8.eg8g _16|:14‾2a&:|‾a_14urq8
630: /B
631: (t9) |:8r1:|
632: /C
633: (t9) q7u+32r8ggg8q6gq8gf8eq6d8.u-16q7dd u+16g8g8q8a8gg&g4r4
634: (t9) q7u-8 r8gggggf8ed8u-8ddq8ue& ed4..r2
635: (t9) q7u+24r8gggg8q8gf8ed8.u-8q7dd u+8g8g8a8u-8gg&g4r4
636: (t9)   u+16r8gggg8gg8g8q8fg8a& q7a4&|:3‾2a&:|‾aq8a4b4
637: /D
638: (t9) |:‾8[$]q7ag8g8.u-16ddu+16aaaaaa8b u-16q8a8.g&g4r4_u+16
b4
639: (t9)   ‾    q7ag8g8.u-16eeu+16aaaaaa8[*]b q8a4r4_a4b4
640: (t9)   ‾    q7ag8g8.u-16d8u+16a8a8a8ab u-16q8a8.g&g4r4_u+16
b4
641: (t9)   ‾    q7ag8g8 u-16eeeu+16aaaaaa8q8b u-16a2r2_16u|
642: /E
643: (t9) u+16q6r8e8q7eg8f8.e8d8>u-16aau+16 b8<dq8d&d4r2
644: (t9) u+16q7r8eeeg8f8.f8u+8a8a8 u+8q8{bag&}4g4r2
645: (t9) u-16q7r8eeeg8f8.e8d8>a8  b8<u-16dq8d&d4r2
646: (t9) u+16q7r8eeee8e8.e8de8. u+16r8gggg8g8.g8eg8‾16a& a1
647: (t9) r2_8a4b4
648: /F(D')
649: (t9) ‾q7ag8g8.u-16q6d8u+16q7aa8a8a8b a8gq8g&g4r4_b4
650: (t9) ‾q7ag8g8.u-16e8u+16a8a8aa8b& ba8.r4q8_a4b4
651: (t9) ‾q7ag8g8.u-16d8u+16a8a8aa8b& ba8g&g4r4_b4
652: (t9) ‾  ag8g8u-16ee8u+16a8a8aaa ‾q8b8a2&_a&_a&_au
653: / u @67 p3 @e44,127 q7 @m0
654: (t9) u+32 ‾4 @30 @p48 @e94,64 @k0 q8
655: (t9) r8r32
656: (t9) >a4.ab<c4.cc# d4.def4.fg a4.ab<c2>
657: /G
658: (t9) [k.sign -b,-e]
659: (t9) |:7r1:|p3‾16@s7@m70r4'a2.<c'
660: (t9) @p95r1r1r1.‾8'<c2e-'@m0_24
661: /I(t2 I とほぼ同じ)
662: (t9) [k.sign -a,-b,-d,-e]‾16
663: (t9) r1...>'a<d''g<c''a8<d'<'df''ce!''d8f''fa''e!g''f8a'
664: (t9)       'a<d''g<c''a8<d'<'df''ce!''d8f''fa''e!g''f8a'
665: (t9)       'a<d''g<c'
666: (t9) 'a<d''fb''g<c''a<d''g<c''fb''e!a''df'
667: (t9) 'e8g''>b<e''df''e8g'|:'eg''fa''gb'&:|@b0,1365'gb'
668: (t9) @b0'gb''fa''df' 'e!g''ce!''d!f''e!g''fa''e!g''fa''gb'
669: (t9) @b-1365,0'a!32<c'&@b0@m30'a!8..<c'&'a!8.<c'&
670: (t9) @b0,-1365@m0'a!<c'_>>
671: /J
672: (t9) v7‾4 u @20 @k-7 o4 p3 @e94,127 q8 r16.
673: (t9) [k.sign b-]
674: (t9) q7rffffffe8du-16c8.cc uf8f8g8fq8f&f4r4
675: (t9) q7r8fffffe8dc8u-16ccq8d8 c8.r2.u
676: (t9) q7rfffffffeedc8.u-16cc u+16f8f8g8fq8f&f4r4
677: (t9) q7u+16‾8r8ffffff8f8df8u+16q8g& g1
678: (t9) r2a4b!4
679: /K
680: (t9) [k.sign +f]
681: (t9) :|
682: /L(E')
683: (t9) u+16q7r8e8eg8f8.e8d8>a8  b8<udq8d&d4r2
684: (t9) u+16q7r8e8eg8f8.u+16a8a8a8  q8‾8ba8g&_g4r2
685: (t9) u-16q7r8eeeg8f8.e8d8>ua8  b8<dq8d&d4r2
686: (t9) u+16q7r8eeee8e8.e8de8. u+16r8gggg8g8.g8eg8‾a& q8a1
687: (t9) r2a4b4
688: /M(D')
689: (t9) ‾8q7a8gg8.u-16q6d8q7u+16a8q6a8q7aa8b a8gq8g&g4r4b8.a
690: (t9)   q7aggg8.u-16e8u+16a8aaaa8b q8a4r4_a4b4‾[d.s.]
691: (t9) [coda]b8q8a4.._b4a4‾
692: (t9)                |:q7ag8g&g4r4|q8b4:|r8u-16e8u+16
693: (t9) <c4cd8c&c4r8>aa< dd8d8d8e8d8.|:q8d8.cq7
694: (t9) c>b8q8b&|b4r4<:|>b2. r1r1
695:
696:
697: /Drums Bass
698: /[Setting]
699: (t10) v12 o2 q7 l16 @1 @u111r1
700:
701: /[Intro]
702: (t10) |:17r1:|r4
703: /AB
704: (t10) |:16r1:|
705: /C
706: (t10) |:|:c4r8.cc8c8r4|c4r8ccr8c8r4:|
707: (t10) |c4r8ccrcrcr4:|c8.crc8.c4r4
708: /D
709: (t10) |:|:c4r4c8c8r4 |c4r8.cc8c8r4:|| |:c8.cr4:|:|c8.cr4ccr
cr4
710: /E
711: (t10) |:4c4r4c8c8r4 c4|r8.cc8c8r4:|r4c8.c32c32r4
712: (t10) c4r8c8r8.c32c32r4 c8c8r4c4r4
713: /F
714: (t10) |:c4r4c8c8r4 | c4r8.cc8c8r4:|c8.cr4c8ccr4
715: (t10) |:c4r4c8c8r4   c4r8.cc8c8r4:|
716: (t10)                c4r8.cc8c8r4
717: (t10) c4r4c8c8r4 c8ccr4c4r4
718: /G
719: (t10) |:4c4r4c8c8r4 | c4r8.cc8c8r4:|c8.cr4c8.c32c32r4
720: /H
721: (t10) |:c4r4c8c8r4 c4r8.c|c8c8r4:|c8.crcrc
722: /I
723: (t10) |:3c4r4c8c8r4 | c4r8.cr8r8r4:|c8.cr4c4r4
724: /J
725: (t10) |:7r1:||:4c4r4:|
726: /D ヲ コピー
727: (t10) |:|:c4r4c8c8r4 |c4r8.cc8c8r4:|| |:c8.cr4:|:|c8.cr4ccr
cr4
728: /L
729: (t10) |: c4r4c8c8r4 c4|r8.cc8c8r4:|r8.cc8.crcrc
730: (t10) |: c4r4c8c8r4 |c4r8.cc8c8r4:|c8.cr4c8.c32c32r4
731: (t10) c4r8c8r8.c32c32r4 c4r4c4r4
732: /M
733: (t10) |:|:c4r4c8c8r4|c4r8.cc8c8r4:|||:c8.cr4:|:|
734: (t10) |:c8.cr4c8.c32c32r4 |
735: (t10) c4r4c8c8r4 c4r8.cc8c8r4 c4r4c8ccr4 :|
736: (t10) |:c4r4c8c8r4| c4r8.cc8c8r4:|c8.cr4c4r4
737: /N
738: (t10) |:c4r4c8c8r4 c4r8.cc8c8r4 c4r4|:3c8.cr4:|
739: (t10) |:3c4r4c8c8r4:| c8.cr4ccrcrcrc
740: (t10) |:c4r4c8c8r4 |c4r8.cc8c8r4:|c8.cr4ccrcr4
```

▶ PICTパズルがいいですね。鉛筆を使うと手が真っ黒,間違って消しゴムを使うとさらに真っ黒でわけがわからなくなるので，これはコンピュータ向きのパズルだと思います。あとは雑誌の問題を入力したら絵が出てくる，自動解法かな。考えてみると，意外と面倒くさいが……。

菅谷　英明(28)兵庫県

```
741: (t10) c4r4c8c8r4 c4r8.cc8c8r4 |:c4r4c8.c32c32r4 |c8.cr4c4r4
¥6
742: (t10) c4r4c8c8r4 c8.cr4ccrc32c32r4:| c8ccrc8.c8.c32c32r4
743: (t10) c4r4c8c8r4 c8.cr4ccrc32c32rcrc
744: (t10) c4r4c8c8r4 c8ccr4c4r4:|
745:
746:
747: /Drums H.H
748: /[Setting]
749: (t11) o2 q7 l16 @u67r1
750:
751: /[Intro]
752: (t11) |:17r1:|r4
753: /AB
754: (t11) |:15r1:|r2 f#a#f#f#f#f#a#8
755: /C
756: (t11) <c#8>|:31f#8:|
757: (t11) <c#8>|:23f#8:| <c#8>f#8r<c#8>rr2
758: (t11) <
759: /D
760: (t11) c#8|:15d#8:|
761: (t11) c#8|:10d#8:| d#rd#rd#8r8d#r
762: (t11) c#8|:15d#8:|
763: (t11) c#8|:10d#8:| d#rd#8d#8c#8d#8
764: /E
765: (t11) c#8|:47d#8:|
766: (t11) c#8|:10d#8:| d#rd#rd#8r8d#r
767: (t11) c#4d#8c#8r8d#8r8d#r
768: (t11) c#8d#8c#rd#rd#r4..
769: /F
770: (t11) |:c#8|:15d#8:||
771: (t11)   c#8|:10d#8:|d#rd#rd#8r8d#r:|
772: (t11) c#8|:5d#8:|r>a#f#<r
773: (t11) c#8|:15d#8:|
774: (t11) c#8|:9d#8:| c#8d#rd#r8.r4
775: /G
776: (t11) c#8|:31d#8:|
777: (t11) c#8|:26d#8:|d#rc#rd#8c#rd#r
778: /H
779: (t11) c#8|:31d#8:|
780: /I
781: (t11) c#8|:31d#8:|
782: (t11) c#8|:7d#8:| c#8d#8c#8d#rc#8d#d#a#4
783: /J
784: (t11) |:7r1:||:4c#4r4:|
785: /D ｦ ｺﾋﾟｰ
786: (t11) c#8|:26d#8:| d#rd#rd#8r8d#r
787: (t11) c#8|:15d#8:|
788: (t11) c#8|:10d#8:| d#rd#8d#8c#8d#8
789: /L
790: (t11) |:4c#8|:15d#8:|:|
791: (t11) c#8d#8c#8c#8|:4d#8:|
792: (t11) |:c#8r4.:|
793: /M
794: (t11) |:c#8|:31d#8:||
795: (t11) |:c#8|:15d#8:|:|:|
796: (t11) c#8|:15d#8:|
797: (t11) c#8|:11d#8:|d#8r4.
798: /N
799: (t11) |:c#8|:15d#8:|
800: (t11)   c#8|:10d#8:|d#d#|:4d#8:|
801: (t11)   c#8|:31d#8:|
802: (t11)|:3c#8|:15d#8:|:|
803: (t11)   c#8|:12d#8:|r4.¥6
804: (t11) |:c#8|:15d#8:| |
805: (t11) |:c#8|:7 d#8:|:|:|
806: (t11)   c#8|:7 d#8:|
807: (t11)   c#8|:4 d#8:|r4.:|
808:
809:
810:
811: /Drums Tom
812: /[Setting]
813: (t12) o2 q7 l16 @u127r1
814:
815: /[Intro]
816: (t12) |:17r1:|r4
817: /AB
818: (t12) |:16r1:|
819: /C
820: (t12) |:7r2.d4:| r4dr8drdu-32<c8>u+16aauff
821: /D
822: (t12) |:6r4d4:| r4d8rdrdr8ddrd
823: (t12) |:6r4d4:| r4d8rdr4d4
824: /E
825: (t12) |:14r4d4:| r4d8rdrdr8ddrd r4d8r4.ddrd
826: (t12) r4ddrd{rddddd}4{aaafff}4
827: /F
828: (t12) |:6r4d4:| r4d8rdrdr8ddrd
829: (t12) |:5r4d4:| r4dr8d
830: (t12) |:6r4d4:| r4d8rd{rddd<cc>}4{aaafff}4
831: /G
832: (t12) |:14r4d4:| r4d8rdrdr8ddrd
833: /H
834: (t12) |:7r4d4:| rdr8d8r8
835: /I
836: (t12) |:10r4d4:| r4d8rdr2
837: /J
838: (t12) |:7r1:||:r4.ff:| r2{raaaaa}4{<cc>aaff}4
839: /D ｦ ｺﾋﾟｰ
840: (t12) |:6r4d4:| r4d8rdrdr8ddrd
841: (t12) |:6r4d4:| r4d8rdr4d4
842: /L
843: (t12) |:7r4d4:| rdr8d8r8
844: (t12) |:6r4d4:| r4d8rdrdr8ddrd
845: (t12) r4d4r4ddrd r8<cc>aara{rddddd}4{aaafff}4
846: /M
847: (t12) |:|:6r4d4:|r4d8rd|rdr8ddrd
848: (t12) |:5r4d4:|r4d8rd r4ddrdrdr8ddrd
849: (t12) |:4r4d4:|r4d8rdr4d8rd rdr8ddrdrdr8ddrd
850: (t12) :|{rd<cc>dd}4{ddaaff}4
851: /N
852: (t12) |:|:5 r4d4:| rdr8ddrd r4d4rdr8ddrd
853: (t12)   |:14r4d4:| r4d8rdr4ddrd
854: (t12)   |:4 r4d4:| r4d8rdrdr8ddrd r4ddrd{rd<cccc>aaafff}2¥6
855: (t12)   r4d4r4d4 r4d8rdr4ddrd |:r4d8rdrdr8ddrd:|
856: (t12)   r4d4r4d4 r4d8rd
857: (t12)   |:3 r4d4:| r4d8rd{rddddaaafff}2:|
858:
859:
860: (p)
```

リスト6 SAY ANYTHING用カウンタ表示

```
 1:00006330 00000000   2:00006330 00000000   3:00008730 00000000   4:00008730 00000000
 5:00005730 00000000   6:00008D30 00000000   7:00008730 00000000   8:00008730 00000000
 9:00006342 00000000  10:00008730 00000000  11:00008730 00000000  12:00008730 00000000
```

リスト7 WAIT FOR SLEEP

日本音楽著作権協会(出)許諾第9571078-501号

```
==================  WAIT_FOR_SLEEP.ZMS  ==================
  1: /----------------------------------------------------------
  2: /          Wait for sleep
  3: /  by  Kevin Moore
  4: /----------------------------------------------------------
  5: .comment  Wait for sleep
  6: (i)
  7: (bl)
  8:
  9: (m1,3000)(amidi1,1)
 10: (m2,3000)(amidi1,2)
 11: (m3,7000)(amidi2,3)
 12: (m4,6000)(amidi3,4)
 13: (m5,3000)(amidi4,5)
 14: /----------------------------------------------------------
 15: .roland_exclusive 16,66{$40,00,$7F,00}
 16: .sc55_chorus{2,0,30}
 17: .sc55_reverb{3,3,0,30,80}
 18: .sc55_v_reserve $10{7,7,6,3,1,0,0,0,0,0,0,0,0,0,0,0}
 19: /----------------------------------------------------------
 20: (o84)
 21:
 22: (t1) @i$41,$10,$42 @e20,5
 23: (t2) @i$41,$10,$42 @e20,5
 24: (t3) @i$41,$10,$42 @e45,60
 25: (t4) @i$41,$10,$42 @e50,30
 26: (t5) @i$41,$10,$42 @e45,60
 27: (t1)@m @q0 @g12@b0@p64@  1@v127     o5@k0  /piano
 28: (t2)@m @q0 @g12@b0@p64@  1@v127@u80 o3@k0  /piano
 29: (t3)@m @q0 @g12@b0@p64@ 50@v127@u65 o5@k0  /strings
 30: (t4)@m @q0 @g12@b0@p64@101@v 95@u95 o3@k0  /vocal
 31: (t5)@m @q0 @g12@b0@p64@ 50@v127@u85 o2@k0  /bass
 32: /----------------------------------------------------------
 33: (t1)
 34: r y$63,$01 y$62,$21 y$06,$4a
 35: t84l16@d1 |:|:3u105b>u75b<u100au105b>u75b:|
 36: u105<<d>u75e<u105c>u75a @d0@d1 |:u105bu75cu100au105bu75ec:|
@d0@d1
 37: u105g>u75a<u105f+g>u75b<u105f+e>u75a<u105d>u75b< @d0@d1 :|
 38: /------[A]
 39: >|:3u95b>u72b<u90ab>u72b:|
 40: u95<<d>u75e<u90c>u75a @d0@d1 |:u95bu72cu90abu72ec:|
 41: @d0@d1 u95g>u75a<u90f+g>u75b<u90f+e>u75a<u90d>u75b< @d0@d1
 42: |:3u90b>u75b<u90ab>u75b:|
 43: u100<<d>u78e<u90c>u78a @d0@d1 |:u90bu75cu90abu75ec:|
 44: @d0@d1 u100g>u74a<u95f+g>u75b<u95f+e8.d8. @d0@d1
 45: /------[B]
 46: u90l8.o4'ceg''ceg''f+d>b''f+d>b'|:4'e>bg':|
 47: 'ceg''ceg''df+a''df+a'|:4'egb':|
 48: 'ce<c''ceb''df+a''dg'|:4'ge>b':|
 49: 'ceg''ceg''df+a''df+a'
 50:
 51: l16@d1 <|:3u100b>u77b<u100ab>u77b:|
 52: u100<<d>u78e<u100c>u78a @d0@d1 |:u100bu77cu100abu77ec:|
 53: @d0@d1 u100g>u77a<u100f+g>u77b<u100f+e>u77a<u100d>u77b @d0@
d1
 54: /------[C]
 55: |:3u90b>u75b<u90abu75e>b:|
 56: u90<<d>u78ge<u90c>u78ae @d0@d1 |:u90bu75cu90abu75ec:|
 57: @d0@d1 u90g>u73a<u92f+g>u76b<u90f+'e>ba'u75du90ef+ga @d0@d1
 58: |:3u90b>u75b<u90abu75e>b:|
 59: u90<<d>u78ge<u90c>u78ae @d0@d1 |:u90bu75cu90abu75ec:|
```

```
 60: @d0@d1 u90g>u73a<u92f+g>u76b<u90f+e8.d8. @d0@d1
 61: /------[D]
 62: t85 l8.o4'ceg''ceg''f+d>b''f+d>b'|:4'e>bg':|
 63: 'ceg''ceg''df+a''df+a'|:4'egb':|
 64: 'ce<c''ceb''df+a''dg'|:4'ge>b':|
 65: 'ceg''ceg''df+a''df+a'
 66:
 67: t84 l16 e>u85e<u94de>u76ae
 68: <u98e>u78e<u102de>u78ae
 69: <u106e>u80e<u109de>u82ae
 70: <u113e>u82e<u116de>u84ae
 71: /------[E]
 72: u106<'d>af+4.'<u120'af+d8'u110'gd8'u100'f+d8'
 73: u100'f+d8.'u93'def+8.'u83>egb<e8.
 74: u95>'egb8.'<d8.>>a8<g8f+8
 75: u90'd>bg8.'u70>e8.u85<'d>bg8.'u70>e8.<
 76:
 77: u100'd>af+4.'<u115'af+8'u105'gd8'u100'f+d8'
 78: 'f+d8.'u93'deg8.'u80r8e>b8.
 79: 'df+a8.'u70>a8.<d8e8f+8'gb8.'e8.'gb8.'e8.
 80: u80'dgb8.'u70<d8.& d8>g8f+8 'ec>g2.'
 81: u80'df+a8.'u70>a8.<'df+8''eg8''f+a8''gb8.'e8.'gb8.'e8.
 82: u90'df+b4.'u95'f+b<d4.''ea<c4.'u100<'df+4.''eg4.''f+a4.'
 83:
 84: t81 @d1 |:3u95b>u78b<u97ab>u78b:|
 85: u95<<d>u78e<u96c>u78a @d0@d1 |:u95bu78cu90abu78ec:|
 86: @d0@d1 u95g>u77a<u90f+g>u77b<u90f+e>u77a<u90d>u77b< @d0@d1
 87: |:3u95b>u77b<u90ab>u77b:|
 88: u95<<d>u77e<u90c>u77a @d0@d1 |:u95bu77cu90abu77ec:|
 89: @d0@d1 u95g>u77a<u90f+g>u77b<u90edegb<df+& f+2. @d0@d1
 90: o4'e>ba2.'
 91: /-----------------------------------------------------------
 92: (t2)
 93: r l16 |:3r4r:|r4|:3r4.:|r4
 94: |:3e<d>b<e8>:| e8<e8
 95: >cea4ce<d4 >>a<eab<def+gf+e
 96: /------[A]
 97: o2u95|:3e8.<e8>:| e4 @d0@d1
 98: c4.c4. a4.@d0@d1 'b<b4'
 99: e2&e8 e8.<e4.
100: c2.>a4.@d0@d1 b4.
101: /------[B]
102: a4.@d0@d1 b4.@d0@d1 e8.<e8.@d0@d1 c4.@d0@d1 >
103: |:a4.@d0@d1 b4.@d0@d1 <e4.@d0@d1 c4.@d0@d1 >:|
104: a4.@d0@d1 b4.@d0@d1
105: u80<|:3e<d>b<e8>:| e8<e8
106: >cea4ce<d4 >>a<eab<def+gf+e
107: /------[C]
108: u95o2e8.<e8.r4.>e8.<e8. r4.
109: >c8.<c8.>r4. a4.b4.
110: o2e8.<e8.r4.>e8.<e8. r4.
111: >c8.<c8.>r4. a4.b4.
112: /------[D]
113: |:3a4.@d0@d1 b4.@d0@d1 <e4.@d0@d1 c4.@d0@d1 >:|
114: a4.@d0@d1 b4.@d0@d1 e4.r4.e4.r4.@d0@d1
115: /------[E]
116: u95'd<d4'&'d<d'<d&d4.>@d0@d1 'c<c4.'c4.@d0@d1
117: >'a<a2.'@d0@d1 <'e<e2.'@d0@d1
118: >'b<b4'&'b<b'<b&b4.>@d0@d1 'a<a4.'a4.@d0@d1
119: <<'d<d2.'@d0@d1 'c<c4.'c4.@d0@d1
120: 'd<d2.'@d0@d1 'c<c2.'@d0@d1
121: 'd<d2.'@d0@d1 'e<e2.'@d0@d1
122: >'b<b2.'@d0@d1 <'c<c2.'r2.@d0@d1
123:
124: u70|:3e<d>b<e8>:| e8<e8
125: >cea4ce<d4 >>a<eab<def+gf+e
126: >|:3e<d>b<e8>:| e8<e8
127: >cea4ce<d4 >>a<et-5at-5b<dt-10ef+t-15gb<t-15dt-15f+t+15b& b
2.
128: o2'eb2.'
129: /--------------- strings ---------------------------------
130: (t3)
131: r y$63,$01 y$62,$21 y$06,$30
132: y$62,$66 y$06,$51
133: y$62,$21 y$06,$30
134: y$62,$64 y$06,$70
135: y$62,$63 y$06,$55
136: l16 |:3'be>b4'&'be>b'&:| 'be>b4'& y$06,$44
137: |:'bec4.'&:| 'gc>a4.''f+d>a4'
138: |:3'be>b4'&'be>b'&:| 'be>b4'& |:'bec4.'&:| 'gc>a4.''f+d>a4'
139: /------[A]
140: >|:3'be>b4'&'be>b'&:| 'be>b4' |:'bec4.'&:|'gc>a4.''ad>a4'
141: |:3'be>b4'&'be>b'&:| 'be>b4'& |:'bec4.'&:|'gc>a4.''f+d>a4.'
142: /------[B]
143: 'ceg4.''df+a4.''egb4.''cegb4.'
144: /'eg<c4.''df+a4.''egb4.''cegb4.'
145: >'a<ega<c4.''b<f+ab<d4.'<'egb4.''cegb4.'
146: /'eg<c4.''f+a<d4.''gb<e2.' 'ceg4.''df+a4.'
147: 'cg<ce4.''>b<f+a<d4.''egb<e4.''cg<ce4.' 'ceg4.''df+a4.'
148: <|:3'be>be4'&'be>be'&:| 'be>be4'& |:'bec>e4.'&:|
149: 'gc>a4.''f+d>a4'
150: /------[C]
151: >|:4'be>b4.'&:||:'bec4.'&:| 'gc>a4.''f+d>a4.'
152: |:4'be>b4.'&:||:'bec4.'&:| 'gc>a4.''f+d>a4.'
153:
154: 'ceg4.''df+a4.''egb2.' 'eg<c4.''df+a4.''egb2.'
155: 'g<ce4.''a<df+4.''gb<e2.' 'g<ce4.''a<df+4.'
156:
157: 'b<e2.'&'b<e2.'
158: /------[E]
159: o4'd>af+4.'<a8g8f+8f+8.'e>bg8.'&'e>bg4.'
160: >'dgb8.''gb<d8.'&'gb<d4.'>'gb<d4.''gb<d4.'
161: o4'd>af+4.'<a8g8f+8f+8.'e>bg8.'&'e>bg4.'
162: >'df+a2.''egb2.''egb8.''gb<d8.'
163: 'gb<d8'g8f+8
164: 'e>gb2.'a4.f+8g8a8'egb2.'
165: @u70'df+b4.''f+b<d4.'@u75
166: <'ce4.'t-3'df+4.'t-3'eg4.'t-2'f+a4.'t85@u63
167:
168: o5|:3'be>be4'&'be>be'&:| 'be>be4'& |:'bec>e4.'&:|
169: 'gc>a4.''f+d>a4'
170: |:3'be>be4'&'be>be'&:| 'be>be4'& |:'bec>e4.'&:| 'gc>a4.'
171: y$62,$68 y$06,$68
172: y$62,$66 y$06,$5a'f+e>baf+4.'&'f+e>baf+2.'
173: >'e>bae2'
174: /-----------------------------------------------------------
175: (t4)
176: r y$63,$01 y$62,$64 y$06,$55
177: y$62,$09 y$06,$55          /depth
178: y$62,$0a y$06,$55          /delay
179: l16 |:|:3r4r:|r4|:3r4.:|r4:|
180: /------[A]
181: r4rb8bbabee4.r r4.b8ab8ag4.r4
182: r4rb8bbabee4.r r4.b8abaag4.r4.
183: /------[B]
184: g8gg8gf+8ee8de8def+g4r8.
185: gggg8f+8.eede4r8.r4.
186: ggggg8f+8ee8de8def+g4r8.
187: g8.g8f+f+8ee8d e8r8.
188: |:r4r:|r4|:3r4.:|r4
189: /------[C]
190: r4.b4bbb8ag8g&g8f+e8e&
191: e8r4|:3r4.:|
192: r4.b8bb8b&b8ag8g&g8f+e8f+
193: ge&e4r4.g8f+e8f+&f+8.r8.
194: /------[D]
195: g8.g8f+e8d&d8de4.r4.
196: b8ag8f+&f+8ga8b&b4.r4.
197: b8bb8<d&(d>b32)&<d.>bb8b<d8ee8e&e8.r8.
198: >b8ag8f+&f+8ed8e&e4.|:3r4.:|
199: /------[E]
200: @u100r4.<d8>b8<d8d8e4r4.
201: d8dd8d&d8>g8f+8e4.r4.
202: r4.<d8>b8<d8d8e4r4.
203: >a4.f+8g8a8b4.r4.
204: b8<dd8d&d8>g8f+8e4.r4.
205: a4.f+8g8a8b4.r4.
206: b8<dd8d&d8>b<d8.e4.r4.
207: /--------------- bass ---------------------------------
208: (t5)
209: r l16 |:4|:3r4r:|r4|:3r4.:|r4:|
210: |:18r2.:|
211: |:3r4r:|r4|:3r4.:|r4
212: |:8r4.:| |:6r2.:|r8
213: l2.ba<dcdc>'d<d''e<e'b<c&c
214: /-----------------------------------------------------------
215: (p)
```

リスト8　WAIT FOR SLEEP用カウンタ表示

```
1:000024E4 00000000   2:000024E4 00000000   3:000024B4 00000000   4:00001F44 00000000
5:00001FD4 00000000
```

リスト9　告白

```
================  LE_TREE.ZMS  ===================
  1: / PCエンジン版「ときめき メモリアル」 (C)KONAMI      /
  2: /
  3: /                    「告白」
  4: /
  5: /                   MUSIC by MEATL-YUKI              /
  6: .comment  OPM :ときめきメモリアル <<    告白    >> (C)KONAMI by ts
ugu.95/5/7
  7:
  8: (i)
  9:
 10: (m1,2000)(aFM1,1)
 11: (m2,2000)(aFM2,2)
 12: (m3,2000)(aFM3,3)
 13: (m4,2000)(aFM4,4)
 14: (m5,2000)(aFM5,5)
 15: (m6,2000)(aFM6,6)
 16: (m7,2000)(aFM7,7)
 17:
 18: /  AR D1R D2R  RR  DL  TL  RS MUL DT1 DT2 AME
 19: (@  1,     22,  8,  8,  8,  5, 42,  0,  2,  3,  0,  0
 20:    31, 15,  8,  8,  8, 40,  2,  4,  3,  0,  0
 21:    21,  8, 12,  8, 12, 62,  0, 12,  7,  0,  0
 22:    19, 10,  8,  8,  8,  0,  0,  4,  7,  0,  0
 23: /  AL  FB
 24:     4,  7)
 25:
 26: / ---------------------------------------------------
 27: (t1) t81 @1v14o5l8p2@q2 @k-4 r*48
 28:
 29: (t1) cd
 30: (t1) |:e4g4f4d4 e2.cd e4g4a-gf4 e2.fg
```

```
31: (t1)  |a4afd4ef g4gec4de f4e4d4c4 e4.{dc}d4cd:|
32: (t1) a4.aa4bb+ ge4<c4>{ge}c4 d4a4g4de t70c2.t+2ct+3dt+6
33:
34: (t1) |:e4g4f4d4 e2.cd e4g4a-gf4 e2.fg
35: (t1)  |a4afd4ef g4gec4de f4e4d4c4 e4.{dc}d4cd:|
36: (t1) a4.aa4bb+ ge4<c4>{ge}c4 t-1d4t-1a4t-1g4t-1dt-1e
37: (t1) t70c*384
38:
39: / ----------------------------------------------------
40: (t2) t81 @1v14o518p1@q2 @k4 r*48
41:
42: (t2) cd
43: (t2) |:|:e4g4f4d4 e2.cd e4g4a-gf4 e2.fg
44: (t2)    |a4afd4ef g4gec4de f4e4d4c4 e4.{dc}d4cd:|
45: (t2)   a4.aa4bb+ ge4<c4>{ge}c4 d4a4g4de |c2.cd:|
46: (t2) c*384
47:
48: / ----------------------------------------------------
49: (t3) @1v9o518p1@q2 @k4 r*47
50:
51: (t3) rr
52: (t3) |:v12c2d4>b4 <c1 c2d4>b4 <c2>v11b2
53: (t3)  |v12a2v11f4v12b4 <c2>v11g4v12b4
54: (t3)   <d4c4>b4v11a4 v12<c2>b2:|
55: (t3) v11a2a4b4 v12<c2e4>v11g4 v12a2v11g4v10d4 e1
56:
57: (t3) o4
58: (t3) |:|:cgeg cafa |:cgeg:|:|
59: (t3)   |:dafa:||:cgeg:|
60: (t3)   |dafa dgfg egfg >b<fef:|
61: (t3) dafa dbgb a2g4d*47 e*384
62:
63: / ----------------------------------------------------
64: (t4) @1v9o518p2@q2 @k-4 r*47
65:
66: (t4) rr
67: (t4) |:v12c2d4>b4 <c1 c2d4>b4 <c2>v11b2
68: (t4)  |v12a2v11f4v12b4 <c2>v11g4v12b4
69: (t4)   <d4c4>b4v11a4 v12<c2>b2:|
70: (t4) v11a2a4b4 v12<c2e4>v11g4 v12a2v11g4v10d4 e1
71:
72: (t4) o4
73: (t4) |:|:cgeg cafa |:cgeg:|:|
74: (t4)   |:dafa:||:cgeg:|
75: (t4)   |dafa dgfg egfg >b<fef:|
76: (t4) dafa dbgb a2g4d*47 e*384
77:
78: / ----------------------------------------------------
79: (t5) @1v9o418p2@q2 @k-4 r*49
80:
81: (t5) v10g4
82: (t5) |:v9c2.v11g4 v10g1 v9c2.v11g4 v10g1
83: (t5)  |v9d2v10d4g4 v9e2v10e4dv9e d4v10c4g4f4
84: (t5)   v10g2g4v11g4:|
85: (t5) v9f2f4f4 g2<c4>e4 f2d4>b4 <c1
86:
87: (t5) o3
88: (t5) |:|:cgeg cafa |:cgeg:|:|
89: (t5)   |:dafa:||:cgeg:|
90: (t5)   |dafa dgfg egfg >b<fef:|
91: (t5) dafa dbgb f2d4>b4< c*384
92:
93: / ----------------------------------------------------
94: (t6) @1v9o418p1@q2 @k4  r*49
95:
96: (t6) v10g4
97: (t6) |:v9c2.v11g4 v10g1 v9c2.v11g4 v10g1
98: (t6)  |v9d2v10d4g4 v9e2v10e4dv9e d4v10c4g4f4
99: (t6)   v10g2g4v11g4:|
100: (t6) v9f2f4f4 g2<c4>e4 f2d4>b4 <c1
101:
102: (t6) o3
103: (t6) |:|:cgeg cafa |:cgeg:|:|
104: (t6)   |:dafa:||:cgeg:|
105: (t6)   |dafa dgfg egfg >b<fef:|
106: (t6) dafa dbgb f2d4>b4< c*384
107:
108: / ----------------------------------------------------
109: (t7) @1v10o518p3@q2 @k0 r*63
110:
111: (t7) cd
112: (t7) |:|:e4g4f4d4 e2.cd e4g4a-gf4 e2.fg
113: (t7)    |a4afd4ef g4gec4de f4e4d4c4 e4.{dc}d4cd:|
114: (t7)   a4.aa4bb+ ge4<c4>{ge}c4 d4a4g4de |c2.cd:|
115: (t7) c*384
116:
117: (p)
```

リスト10　告白用カウンタ表示

```
1:00001920 00000000   2:00001920 00000000   3:000019DE 00000000   4:000019DE 00000000
5:000019E1 00000000   6:000019E1 00000000   7:0000192F 00000000
```

(善)の「勝負はこれからだ」

ロンドンの話

観光で15年ぶりにイギリス，ロンドンを訪れた。その昔住んでいたことがあるのだが，あの保守的な国がずいぶんと近代化されていたのに驚いた。地下鉄のホームには電光掲示板があったしエスカレータも木製からステンレス製になっていた。しかしゲームセンターは並んでいるものは新しいものばかりだが雰囲気の暗さとメンテの悪さは昔のままだった。また，昔ながらのスロットマシンやコイン落とし，ピンボールの設置台数のほうが圧倒的にビデオゲームを上回っており，国民性の違いを物語っていた。もっともスロットマシンやコイン落としなどのコイン系ゲームはすべて現金が直接出てくるのでもっともインカムの多い台といえる。設置台数が多いのも納得のいくところではあるが(ちなみにビデオゲームは1プレイ£0.5＝約75円で缶ジュースもだいたいこの値段)。

とはいえ，バーチャファイター2(SEGA)，ストゼロ(CAPCOM,海外名:STREET FIGHTER Alpha)，THE KING OF FIGHTERS '95(SNK)，Killer Instinct(任天堂)などの対戦格闘ゲームはやはり人気がありロンドン子が大勢群がっていた。

対戦台 in London

まず，対戦台は日本でメジャーな向かい合うタイプはほとんどないのが印象的だった。隣同士に座ってプレイするタイプが主流なのだ。皆，気軽に「May I join in?」と声をかけ「Sure!」の答えを聞いて乱入していく。そして勝負がつくと日本じゃ変人扱いされるほどのオーバーアクションで自分をアピールする。

店員が怠慢なのか，ほとんどのゲームにインスト(説明)カードが貼られてない。ボタンやレバーの調子が悪い。この辺は15年前と変わらなかった。Leicester Squareのある場所では小キックと中パンチしか利かないストAlpha台で現地人が対戦で盛り上がっており，西洋人の一面を垣間見た気がした。うーむ，大雑把。

さて私も7/28現在日本でも出たばかりのSNK，KOF'95の乱入台に挑戦してみることにした。場所はロンドンの最中心部Piccadilly Circusだ。私自身プレイするのは数回目だが同'94は相当やり込んだのでそれなりに腕に覚えはあった。相手はちょっと顎ヒゲを生やした陽気な黒人。どうせレバーをガチャガチャやってわめくだけだろう，日本人ゲーマーの腕前をとくとご覧あれ……ありゃりゃ，結果は惨敗。相手の黒人は確かにオーバーアクションでプレイしてはいるが，その操作は正確無比。いわゆるキャンセル連続ワザをバシバシ使い，フェイントなども織り交ぜたりして小癪，日本の名人級なのだ。そのあと別の白人青年にも挑戦したが同様にうまく，結局勝つことができなかった。

現地人にいくつかの質問をぶつけてみた。

「なぜここ(Piccadilly Circus)の連中はレベルが高いの？　Leicester Squareでは結構みんな大雑把なようだったけれど」

「ここはロンドンで一番ゲーム好きが集まるところだからさ。新作も一番早く入荷するしね」(ほぼ日本と同時に発売されるらしい)

「日本のAmusement Spaceではすべての台に説明が付けられているのだけど，なぜロンドンでは説明カードがないのにみんな上手なの？」

「みんな独自の情報ネットワークを持っているのさ。大学生はインターネットとかね。でも一番みんながやっているのは日本のゲーム攻略本を買ってきて見ていることだね」

確かにロンドンでは日本の本/雑誌を売る本屋がいくつかある。St.Paul's寺院の裏にある店は私が住んでいたときからあった。どうやらこちらのゲーマーはこういうお店で攻略本を買っているらしいのだ。なるほど日本語が読めなくても写真や図が多い「ゲーメスト」などならば確かに眺めるだけで内容がわかるかもしれない。なお，現地ではパソコン雑誌はかなり種類があるがゲーム攻略雑誌はほとんどない。ファミコン系ならば少しはあるようだが，アーケードゲームの攻略というような記事にありつくには日本の雑誌を見るしかないという状況らしい。

「みんなこういうゲームが日本製だっていうことは知っているのかな」

「もちろんさ。だからロンドン中でいちばん日本人が親切にされるのはAmusement Spaceかもしれないね(笑)」

(気が向けば続く)

(善)のゲームミュージックでバビンチョ

西川善司

今月は夏休みスペシャルってことで2ページだ。次世代機(もう登場しているんだから新世代機とでもいうべきか？)が発売されてから，最近またにわかにゲームミュージックが面白くなりだした。さぁ今月はその次世代機用からいってみよう。

●アーク ザ ラッド
――オリジナルゲームサントラ――
CD：ARCJ12　2,800円(税込み)
アンティノスレコード　発売中

Play Station待望の本格ロールプレイングとして登場したこのゲーム，売上本数はかなりのヒットを記録したが，内容に対する評価については，世間は意外に厳しかったようだ。BGMに関してはT-SQUAREの安藤まさひろが担当，しかも初挑戦ということでゲーム発売前より話題となっていたが，はたしてそのデキやいかに？

オープニングソングは主題のはっきりした映画音楽調交響曲。ブラスが刻むリズミカルなフレーズに乗って「インディ・ジョーンズ」を思わせる勇ましい旋律が走る。なかなかかっこいいぞーと聞いているうちにオーケストラ曲が感動のうちに終わる。と，あとはエンディング曲まですべてモロT-SQUAREフュージョンが延々と続く。

まるでこれではゲームミュージックというよりはT-SQUAREのアルバムではないかーとライナーノートに目をやる。確かに作曲は全曲安藤まさひろだが，オーケストラ調のものは日本映画音楽で著名な奥慶一がアレンジを担当している。そして一方，T-SQUAREキーボードの和泉宏隆がアレンジを担当している曲はすべてT-SQサウンドになっているようだ。オーケストラ曲は演奏をロイヤルフィルハーモーニーオーケストラが担当。それじゃ「T-SQ」サウンドのほうは？　と見てみるとベース須藤満とドラム則竹裕之，サックス本多雅人……まるごとT-SQUAREのメンバー……。

ゲームをまったく知らない，またはゲームミュージックを聞かない人でもフュージョンが好きなら満足できる内容と断言できる(これってほめてることになるのかな)。

・おすすめ度　10

●モータートゥーングランプリ
――オリジナルゲームサントラ――
CD：ARCJ16　2,000円(税込み)
アンティノスレコード　発売中

昨年末に飛び出したSCE発売のPS用コミカルカーレーシングゲーム「モータートゥーングランプリ」のオリジナルサントラアルバム。戸田誠司と永田英哉作曲の元気一杯のサウンドが目白押し。

戸田誠司の曲はメインテーマとトゥーンアイランドの曲がいい。特に後者は，バンジョーの高速バックアルペジオとブラス隊の緊張感のある旋律がレースゲームならではのスピード感を醸し出している。

永田英哉はもはやX68000ユーザーでは知らない人がいないというほどの有名なゲームミュージシャン。永田氏の曲の中では特にオーケストラ調のガリバーハウスコースの曲がいい。チャイコフスキー「白鳥の湖」(4羽の白鳥の踊り)を彷彿とさせるバスーンの伴奏にオーボエの旋律という構成から，徐々にさまざまな楽器たちが演奏に参加してくる展開は音楽的に大変素晴らしい。

2曲ほど東京パフォーマンスドールの八木田麻衣という女の子が音痴な歌を披露してくれるが，まぁこれは我慢しようか。

・おすすめ度　9

●エアーコンバット22
VHS：VIVL-149　4,900円(税込み)
LD：VILL-103　4,900円(税込み)
ビクターエンタテインメント　発売中

ナムコの誇るシステムスーパー22の最新CG性能を駆使して，よりリアルな空中戦が楽しめるようになった「エアーコンバット」の続編「エアーコンバット22」。今回その映像をすべて収録したビデオ/LDが発売となった。

僚機の配備，対地攻撃や対艦船攻撃などの作戦内容の多彩化……ゲーム内容的にも前作と比べて豊富になり，したがってその映像もより魅力あるものになっている。

見所はもちろんその迫力あるドッグファイトなのだが，システムスーパー22が作り出すそのリアルな映像全体に目を配って見ていただきたい。山岳の岩肌や田園地帯の地形がかなり遠方まで見え，さらに遠方に行けば行くほど霞がかって見えている。それら地形が遥か下方をゆっくりと後ろへ流れていき，そして自機の旋回によってさまざまな様相を見せる。フライトシミュレータの醍醐味が満喫できるこのビデオ，ゲームファンならずとも一見の価値あり。

ひとつ気になった点。私はこのエアーコンバット22は空中戦ゲームということで，敵をロックオン→ミサイル発射→ズガーンという図式を期待して見ていたのだが，ビデオ中のゲームプレイヤーは得点稼ぎなのかなんなのか知らないが，ほとんどミサイルを使わない。ゲーセンでゲームしてんじゃないんだからもうちょっと見るほうの立場になって映像作りをしてもらいたいものだ。

・おすすめ度　9

●ダライアス外伝
VHS：PCVP-11702　4,800円(税込み)
ポニーキャニオン　8/19発売

「レイフォース」に続くZUNTATA自らプロデュースを手がけた映像作品(?)が発売。前回のレイフォースでは女性型生態兵器の記憶を再生という設定で作られていた。今回はあまり「ダライアス外伝」とは関係のないドラマとダライアスのゲーム映像を強引にシンクロさせて綴っている。

ドラマのほうは二重人格の少女とパソコンマニアの精神科医師，そして少女に恋するゲーム好き少年と，謎の殺人鬼が登場し，サイキックスリラー的なストーリーを見せてくれる。しかしかなり意味不明にもかかわらず，ビデオの収録時間のかなりの部分を占めており，これはちょっと賛否両論発生の兆し。ゲーム映像の収録時間の関係なのかはわからないが，ビデオ後半では画面を4つに分割して一度に4ステージを紹介するというのはチョット……ねぇ。

ダライアス外伝

ダライアス外伝の映像を紹介したかったのかドラマを見せたかったのか。ちょっと謎である。

・おすすめ度　7

●MIDI POWER Pro〈ベストセレクション〉
CD：KICA7670　3,800円(税込み)
キングレコード　発売中

MIDI POWERシリーズといえばコナミの歴代のゲームミュージックをRolandのGS音源用にアレンジ，そして演奏して収録したモノ。しかし単なるアレンジバージョンという位置付けで片づけられないのは，その演奏のすべてが，1台のアマチュアでも使える機材で行われている点だ。本シリーズはコンピュータミュージックファンのお手本的な存在でもあるのだ。

今回発売されたベストセレクションは，いままで5枚発売されている同シリーズから人気のある曲を選りすぐってまとめたもの。しかし普通のベストアルバムと大きく違うのはパッケージに1枚のFDが付属しているところ。このベストセレクションには収録に使われた演奏データと同一の演奏データがスタンダードMIDIファイル（SMF）として付録FDに収められているのだ。すなわち，手持ちのコンピュータのSMFプレイヤでCDと同じ演奏が生(!?)で自分の家のGS音源で楽しめるというわけだ。

ちなみにFDは3.5インチ(1.44Mバイトフォーマット)なのでX68000ユーザーは自分のマシン環境でこれが利用できるかどうか，よく確かめてから購入すること。

・おすすめ度　10

●THE KING OF FIGHTERS '95
CD：PCCB-00187　1,500円(税込み)
ポニーキャニオン　8/19発売

去年最大のヒットを記録した対戦格闘ゲーム「THE KING OF FIGHTERS '94」の続編「～'95」が1年ぶりに登場。SNK人気キャラ総登場の格闘ゲームということで今年もヒットの予感が。そして早くもオリジナルサントラの発売が決定した。

新キャラの追加のみならずステージが一新されたこともあって曲はほぼ全曲新曲。'94のときのアンディやアテナのテーマのように昔の曲をアレンジ，というパターンも予想に反して1曲もない（Rのテーマはあるけど）。ただし各キャラ(および登場ゲーム）のイメージはそのまま受け継がれている。気にいった曲はトラック4の日本ステージのテーマとトラック12のアメリカステージのテーマ。特に後者はバックのリズミカルなカッティングギターとオルガンコードにのせて狙ってんだか狙ってないんだかわからないアンニュイなサックスリードがおしゃれ。展開もいい。

このCDを買うと例によってボスや隠れキャラの謎が解けるかも。だからSNKは商売うまいって。

・おすすめ度　7

●サイバーボッツ
――アーケードゲームトラック――
CD：SRCL3274　3,200円(税込み)
ソニーレコード　発売中

類似品が氾濫する格闘ゲーム界に一石を投じるべく登場したロボット対戦格闘ゲーム「サイバーボッツ」。しかしフタを開けてみれば，操作系の一新やキャラごとのフィーチャーはあるものの，本質的にリュウやケンがロボットに置き変わっただけのストIIとなんら変わらないものだった。

しかし音楽はかなり個性的。同社の「ヴァンパイアハンター」ではディスコやダンスミュージックを基調とした曲作りがなされていたが，今回の「サイバーボッツ」ではハードロックを基調としているようでどの曲もアグレッシヴなギターサウンドを前面に押し出している。とはいえ，ツーバスでドキャドキャやっているクレージーな曲はなく，このあたりはゲームミュージックという枠に収まっているので安心といえる(!?)。

・おすすめ度　7

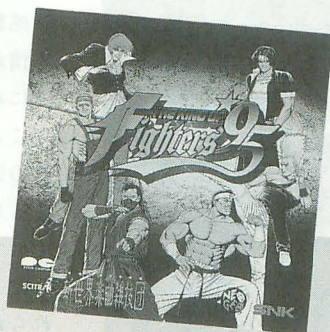
●ミュージックフロム「風の伝説ザナドゥ」
CD：KICA-1165～6　3,800円(税込み)
キングレコード　8/23発売

PCエンジン用のRPG「風の伝説ザナドゥ」のCD2枚組のオリジナルサントラ＋α。PSG音源とCD-ROM音源の2種類の音源で演奏される76曲もの楽曲がすべて収録されている。うち5曲が今回の収録のためにさらにアレンジされ追加収録されている。

今回聞いたサンプルはCD-ROM音源のもののみなのだが，こちらはとにかく素晴らしかった。全曲オーケストラ構成で演奏される壮大なアンサンブルはまさに感動もの。サウンド的にも音楽的にも優れている。

ところで，叙述的音楽というものは大きく分けると2種類ある。ひとつは表現したいものがそこにありその情景を音により効果音的に修飾するタイプのもの。そしてもうひとつは，表現したいものそのものをキャンバスに絵の具を塗るように音によって表現するタイプのものだ。そしてこの「風の～」は後者のタイプ。ゲームに対する知識がまったくなくても，聞いているうちに物語，登場人物などの感情までが自とわかってくるような写実的で視覚神経に訴える内容だ。聞いているとどんどん引き込まれていく感じさえする。後者のタイプの管弦楽を特に交響詩などといったりするが風の伝説ザナドゥはまさにこれである。

・おすすめ度(DISC2に関して)　10

MIDI POWER Pro

THE KING OF FIGHTERS '95

風の伝説ザナドゥ

アマチュアCGA現状論(後編)

プロジェクトチームDōGA
かまた ゆたか

今月は，先月に引き続きアマチュアCGAにおけるテーマやストーリーについての私的な考察と，久々のアマチュアCGA学会レポートとして，3D電子遊戯が苦手な人のための実験をお届けします。

はじめに

まずはじめに業務連絡ですが，CGAコンテストのビデオの発送は，なんのトラブルもなく，終了いたしました。万が一，申し込んだのにまだ届いていないという方がいらっしゃいましたら，振り込み用紙のコピーを添えて，ご連絡ください。

さて，アマチュアのCGA作品が発表されるようになって，もうすぐ10年が経ち，技術は格段に進歩したものの，内容も大きく向上しているといえるかは疑問です。いまこそ内容，つまりテーマやストーリーというものについて考える必要があるのではないでしょうか。

テーマやストーリーは，どんなコンテストにおいても，審査に大きく影響する重要なポイントです。また，プロのCGA作家の場合，テーマやストーリーは，クライアントやディレクターが勝手に決めてしまうことが多いのに対して，アマチュアは自由です。まさに，アマチュアの特典といえます。

8月号では，CGA作品を制作するうえで，テーマだとかオリジナリティということを考察してみました。今回も引き続きテーマについて考えるとともに，ストーリーの作り方についてよい方法を探ってみましょう。CGA作品を作りたいのはやまやまだが，ストーリーを思いつかないという方は参考にしてください。

歩行者とドライバーのコミュニケーション

またコラムでは，久々のアマチュアCGA学会のレポートとして，新世代ゲーム機の登場でたくさん出現した3Dゲームに関する実験をお届けします。

感動型テーマの原則

8月号では意見型のテーマを取り上げました。しかし，意見型のテーマは，あまりテーマをストレートに押し出しすぎると「歩行者とドライバーのコミュニケーション」(太田敦司さん)のように，プレゼンテーションになってしまいます。プレゼンテーションが悪いというわけではありませんが，アマチュアCGAの王道とはいえないでしょう。

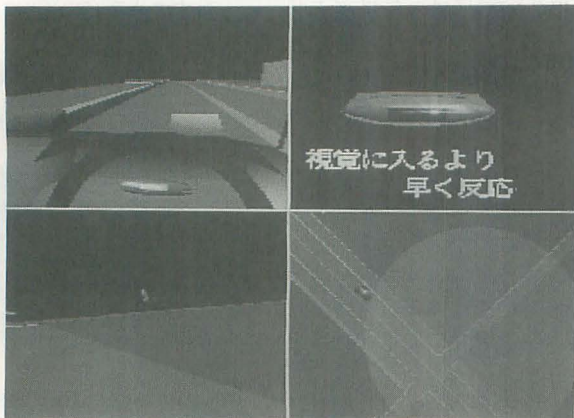
その点，感動型のテーマは，まさにアマチュアCGAの王道といえます。感動型のテーマの作品は，とにかく自分が体験した感動を，映像を媒体に視聴者に伝えることを目的にしており，見終わった視聴者が感動すれば成功，感動しなければ失敗です。

“感動”というと，なにか思わず涙を流さずにはいられないような強烈な出来事のように思われますが，残念ながらそんな体験は誰にでも頻繁にあるわけではありません。ですからそんな大げさなものと考えずに，文字どおり“感情が動く(変化する)”と考えてみてください。

腹が立ったこと，面白かったこと，なんとなく悲しかったこと，悔しかったこと……。その程度なら，だれでも1日に数回はあります。そして，その程度の感動で十分作品は作れるのです。

>感動型テーマの原則
感情が変化するようなことがあれば，それはすべてテーマになりうる

ここでポイントになるのは密度，つまり単位時間あたりの感動量です。日常のごく小さな感動をテーマにすることに問題はありませんが，それを2時間かけて表現していては，また“内容がない”といわれてしまいます。

たとえば「Low Reso」(中村ゆういちさん)では，“忘れられない初恋のときめき”をなかなかうまく描いていますが，一部の審査員には，あれだけの時間をかけるほどではないと減点されました。

この場合の対策は3通り考えられます。ひとつは，作品を短く簡潔にする。2つ目に小さな感動を複数盛り込む。そして3つ目は，大きな感動にする。どの策がベストかといえば，なんといっても短くすることでしょう。少なくとも無理に話を壮大にして，大きな感動にしようという方法だけは賛同いたしかねます。

テーマは身近で

劇場映画などを見ていると，ほとんどがきわめて数奇な事件や大きな感動を描いています。ですからCGA作品を考えるときも，せっかく手間をかけて作るのだからといって，なにかものすごい感動を描こうとしがちですが，これはいけません。劇場映画とアマチュアCGAはまったく別物で，描くもの(テーマ)も描き方も違うということを念頭に置くべきです。

日常の身近な話からテーマを得るほうがよいというのには，いくつか理由があります。まず，誰にでも思い当たることのほうが，当然のことながら共感しやすいというメリットがあります。これは，テーマの定義，作品制作の目的を考えてもきわめて重要なポイントです。

次に，身近な話でないと，その世界観，状況設定を説明するだけでも手間がかかってしまいます。アマチュアCGA作品は長くて15分，通常は5分以内です。この点が劇場映画と最も異なる点です。ひとつの方法としては，一般によく知られている世界観，状況，たとえば「STAR WARS」「ガンダム」「グラディウス」などから拝借すると説明を省略できますが，これではオリジナリティがないとか，どこかで見たような作品だとかいわれてしまいます。

最後に，身近でない世界だと，どうしても空想に頼りがちになってしまいます。プロやかなり手慣れた人ならともかく，普通の人が空想に頼れば，どんどんリアリティが欠如してしまいます。まぁ，CGA作品の場合，リアリティはあまり追求しませんが，説得力がなくなるのが問題です。

過去のCGAコンテストを振り返ると，日常から題材をとった作品は意外と少ないことがわかります。第5回の「ある夜の出来事」(小島禎樹さん)ぐらいでしょうか。私はこの作品の内容を高く評価していて，賞は十分取れると思っていたのですが，ちょっと絵が足を引っ張っていたように思います。

そこで，日常からテーマを見つけるために，日頃からとにかく自分の心に変化があるようなことがあれば，テーマになるかもしれないので一度はチェックする癖をつけるようにします。

>テーマを探すための標語

これはテーマになるんじゃないか

その体験が，そのまま，そのひとつだけで作品になるかといえば，そんなことはめったにありません。しかし，ヒントやネタになるようなことは，たくさん転がっています。そういったアイデアは忘れないうちにメモでもとっておきましょう。

CMを見る

先ほどもCGAは劇場映画とは違うメディアだと述べましたが，意外と共通点が多いのがテレビのCMです。もちろんCMのほとんどは，ただ商品名を連呼するだけのものや，商品の特徴を解説するようなものですが，この類は参考になりません。そのなかで，お酒やタバコなど，イメージを売り込むタイプのCMは，きわめて短い時間の映像でイメージを伝えるということで，CGA作品のお手本になります。

思いつくものをいくつか具体的に挙げてみましょう。

商品名：サントリーオールド

・混んだバスの中。30歳代の女性が降りようとしているが，苦労している。

・本を読んでいた男性がひと言「通してやれよ」と声をかける。

・女性が，振り返る(アップ)。

・バスの外。バスが動き出す。女性，バスの中を見る。

・男性は，なにごともなかったように本を読んでいる。

・バスが走り去る。

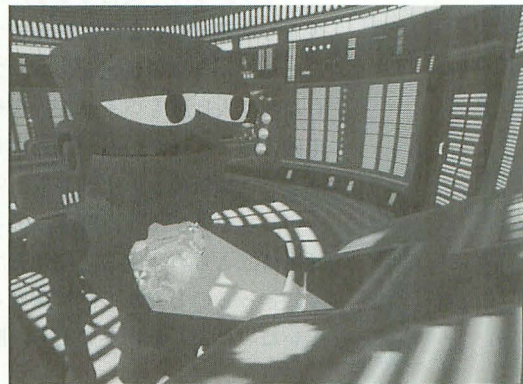
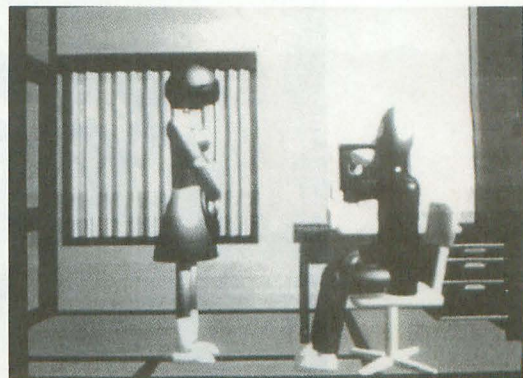
・女性，なんとなくうれしそうに立ち去っていく。

・テロップ：恋は，遠い日の花火ではない。

私の文章ではなかなか伝わりにくいとは思いますが，見知らぬ人に親切にしてもらって嬉しかった，ちょっとすてきな異性を見てときめいたという程度のごく日常の1シーンながら，恋は若者の特権ではないというテーマを描き，さわやかな感動があります。

さらに，ちょっと古いCMを紹介しましょう。

商品名：ポカリスエット

・若い女性（一色沙英）が炎天下，屋根に白いペンキを塗っ

Low Reso

ある夜の出来事

ている。まだ半分も塗れていない。
・友人らしい男女が数人，自転車で通りかかる。下から声をかけて誘うが，無理だとわかって立ち去っていく(すべてロングで台詞もなし)。
・汗をかきながら，ペンキを塗り続ける。
・ハシゴが掛けてあるところに，ひょっこりと冷たそうなポカリスエットが2つ置かれる(アップ)。
・塗り終えた白い屋根の上で，2人がポカリスエットを飲んでいる。
・青い空が広がっている。

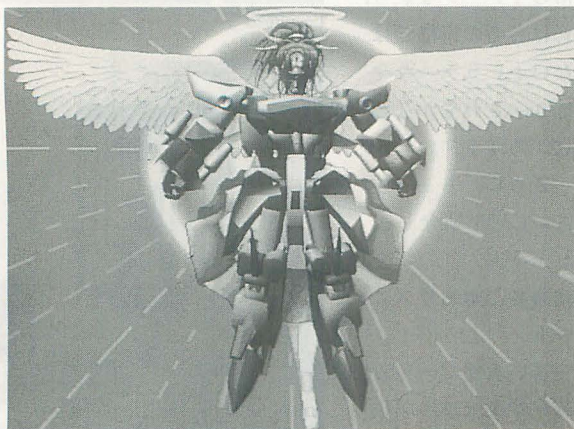
これもたいへんよくできた映像ですね。短い時間に夏のイメージや，青春の1ページをうまく描いています。よく10数秒に詰め込めたものです。

例を挙げているときりがありません。皆さんもこれからテレビを見るときは，CMも気を抜かずにチェックしてみてください。

日常からストーリーを生み出す

テーマもネタも日常で見つかるといわれても，自分の日々の生活を振り返ると，どう考えても作品になるとは思えないという人も多いと思います。しかし，その考え方が最大の問題です。最初からないと思い込んでいると，見過ごしてしまうものです。

たとえていうなら，映画に出てくる主人公の生きざまは宝石で，我々の日常生活は自分の足元の土砂のようなものです。なんの変哲もなく，周りの人たちと同じで，少しも価値がないように思えます。

しかし，自分の足元をスコップで掘ると金銀財宝だらけなんて人は誰もいません。金や宝石の原石は，普通の土砂の中に，ほかの石や砂と混じっているのです。そのままでは価値のない土砂でも，その中から一粒一粒光るものだけを選び出し，磨き，あるいは溶かして再結晶させれば，きらめく宝石となります。あなたの足元からでも，きっとひと握りの金が採れるはずです。ポイントは，ていねいにより分けることと，磨くことです。

電神ギガダイン

まずは，先にも述べたように，常日頃から，なにか自分の感情に変化があったとき，“これはネタになるんじゃないか”と検討してみます。そして，その場ではなにもストーリーが思いつかなくても，ある程度有効性がありそうならば，心の片隅にメモしておきます。質はともかくとして，まずは量を稼ぐわけです。

次に，これはそこそこいけるんじゃないかと思われるネタが見つかれば，そのネタに類似する感情の変化をもった過去のネタを集めてみます。たとえば，“飼っていた犬が死んだ。特に自分がかわいがっていたわけではないが，むしろそのためか少し罪悪感を伴った悲しさを感じた”ということがあれば，“罪悪感”を感じたことや“悲しかった”ことを過去のライブラリから検索します。この場合，“犬”“ペット”“死”というキーワードで検索してもよいでしょう。

1）親の財布からお金をちょろまかした。
2）友人と映画を見に行く約束をしていたのに，すっぽかされた。
3）子犬のころ川へ連れていくと，汚いどぶ川で泳ぎだして臭かった。
4）台風の日の晩にカナリアが死んだ。母はこれはみんなの身代わりだといった。
5）祖母の葬式に出て，祖母の顔を見ると，白っぽく不気味で怖かった。

こんな感じで，いろいろ出てきます。

そして，検索の結果を総当たり的に組み合わせてみます。大抵のものは，どう考えてもつながりようがないでしょうが，なかには関連性があるものも見つかるはずです。関連性が見つかれば，またさらにその関連する事項で検索します。そうしていくと，なにか一本の線でつながりそうなネタの集合体ができてきます。

このようにネタを膨らませる半面，そぎ落とす作業も重要です。この線で行こうというのが大体見えてきたら，そのラインから外れるような部分は，どんなによいネタであっても外していかなければいけません。あまり欲張って，いろいろ盛り込むと，結局なにがいいたいのかよくわからなくなってしまいます。第7回の「電神ギガダイン」(腰原仁志さん)もその例といえます。

また，それぞれのネタについても，状況を整理して，そのネタのいちばんのポイントを生かすために，冗長な部分は省略し，より効果的な状況にします。この段階で初めて創作，つまりフィクションの要素が入ってきます。たとえば，上の5)のネタと関連させる場合，もともとその犬は妹が拾ってきたとしても，祖母からもらったことにするとか，祖母が死んだ直後にあとを追うように死んだとか，都合のいいようにでっち上げるのです。

このように，関連あるネタを組み合わせ，順番につなぎ合わせて，空白の部分は創作すればストーリーができてきます。すんなりできればそれに越したことはありま

せんが，うまくつながらないとか，空白の部分が埋まらないとか，創作の部分がほとんどを占めてしまったとかいうこともよくあるでしょう。その場合，構想中ということで，アイデアノートに途中まででよいので記録しておきましょう。私もそういった構想中のストーリーが20～30あります。

そしてまた常日頃，なにかことある度に，これは構想中のどれかに利用できないかなとチェックしていくわけです。

ストーリー作成のヒント

以上の話で“なんとなく自分でもストーリーが作れそうだなぁ”という気になっていただければ幸いです。そのほか，ストーリー作りに役立ちそうな話を簡単にまとめます。

1）夢からネタを得る

まず，ストーリーを作るためのネタですが，まず自分が実際に体験したことを中心に組み立てるという話はすでにしました。一から話を作ると，嘘くさく，どこかで聞いたような話になってしまいます。

そんなとき意外とネタになるのが，夢です。将来の夢ではなく，寝ているときに見るほうの夢です。多くの場合，夢を見ているときは，それが夢なのか現実なのかわかりませんので，本人にとっては実体験と同じような経験になります。

ただ，多くの夢は矛盾だらけで，そのままではわけがわかりません。ですが，意味があるように差し替えたり，修正したりすれば，なかなか奇抜なネタとなることがよくあります。

もうひとつの問題として，せっかく見た夢も朝になったら忘れているということがあります。しかし，ちゃんと覚えておくにはコツがあるのです。朝，起き上がる前に，体を動かさない状態で，頭の中で夢の内容をなんども反芻し，できれば言葉に直します。そして，あらかじめ枕元にノートを用意しておいて，起き上がるやいなや，そのノートに記録します。私は，この方法で，多くのネタを夢から得ることに成功しているのですが，ほかの人にもできるでしょうか？　いきなりできるようになるとは思いませんので，気長に挑戦してみてください。

2）BGMでまとめる

次に，ネタもいろいろあり，なかには脈絡のありそうなものもいくつかあるが，なかなかひとつのストーリーには至らないという場合には，よい方法があります。まず，BGMを決めるのです。

たくさん曲を聞いて，中心にしたいネタのイメージに合う曲を見つけます。そしてその曲を聞きながら，ネタの推敲や，創作をします。すると，曲の変化に合わせて，ストーリーが浮かんできますし，全体として統一性のある内容になります。

しかしながら，著作権上，その曲はそのまま使えないのが難点です。曲は知人が作ってくれるという場合，それに似た曲を作ってと頼んでも，微妙にイメージが違うとむしろ気になりますので，きっぱりとあきらめ，映像だけを見せて，それに合う曲を作ってもらうほうがよいでしょう。

3）2部構成

ストーリーの基本が，起承転結というのはすでにご存じだと思います。しかし，ストーリーを考え出すとついつい長い話になってしまうという人(たぶん映画の影響が抜けない人)は，この起承転結の構成をやめて，起・結という2部構成で考えてみてはいかがでしょうか。

この場合，「結」は感動で，「起」はその原因です。ある感動，つまり感情の変化と，その感情の変化が起こった理由だけという非常にシンプルな構成です。それ以外の話はばっさり削除し，絶対に語らないと話がつながらない部分だけを残します。

また，ギャグの場合，まずオチを考えて，そのオチに至る話をでっち上げるという手法もあります。

演出

“演出”というのも，なんだかわかるようなわからないような曖昧な言葉です。まずは，明確に定義してみましょう。

>演出の定義

演出とは，わかりやすくするための工夫

なにをわかりやすくするかといえば，たとえば凝った演出で雰囲気を出せば，その状況をわかりやすくしたり，ストーリーをわかりやすくしているといえるでしょう。でも，それらは最終的にテーマを語るための手段ですから，結局テーマをわかりやすくするための工夫であるといえます。

ですから，その工夫によって，ストーリーやテーマがわかりやすくなっていれば，それはよい演出です。逆にわかりやすくなっていなければ，よけいな演出だといえます。

たとえば「電神 ギガダイン」では，主人公と各ロボットの関係が，単なる操縦者と機械という関係にならないように，話しかけるときに“みんな”とか“いつも○○っていっているでしょ”のように人に話しかけるような口調になっています。この作品では，人間とロボットという種別を超えた仲間を描くのがテーマですので，なかなか重要な演出だといえます。その半面，敵キャラのボスが，妙にカッコよく描かれているので，テーマがよくわからなくなってしまったという感じもします。これは過剰な演出といえるでしょう。

では，具体的にどんな演出があるかという話を始める

ときりがありません。オープニングにしろ，タイトルの出し方にしろ，工夫の仕方など各カットごとに無限にあります。今回は，トータルの演出，すなわち構成のポイントだけ箇条書きにしてみました。

1）話の順番を工夫する。ストーリーがわかりやすいように，感動の部分が盛り上がるように。起承転結。

2）よけいなものを徹底的に削る。入れなくても話がわかればないほうがよい。

3）冒頭で，できるだけ速やかに最小限必要な状況を理解させる。

4）始まったら，できるだけ早く本題に入る。

5）興味(謎など)を持続させる。

6）テーマを主人公の台詞でストレートに語らない。

7）オチの部分を明確にする。

8）オチがついたり，本題が終わったら，早く終わる。

それぞれ，例を挙げて解説してもよいのですが，いたずらに長くなりそうなので省略します。

幸い演出は，ストーリーやテーマと違い，ほかの作品から得たものであってもあまり非難されません。ですから，積極的に映画やアニメを注意深く見て，いろいろな演出を盗むべきです。ただ，そうやって演出法を身につけるのはよいのですが，実際の作品制作に生かす場合は，単にこんな方法を知っているぞ，思いついたぞというような興味本位の動機で使用するのではなく，それを用いることで，本当にわかりやすくなっているか，つまり必然性を確かめるようにしてください。

現在のアマチュアCGAは，歴史が浅いせいか，構想が短いせいか，演出がまだまだ弱いといえるでしょう。逆にいえば，そこを強化することで，皆さんの作品もまだまだ伸びるということです。

おわりに

前回と今回の2回に分けて，アマチュアCGAにもっと内容をつけるためにはどうすればよいだろうかという，雑談めいたことを申し上げました。簡単にまとめると，以下のようになると思います。

・いろいろなことを実際に行い，経験を得なければならない。

・なにごとにも興味をもち，ほかの人がもっていないような知識を得なければならない。

・ものごとに対して疑問をもち，ほかの人の意見を聞くと同時に，自分なりの意見をもたなければいけない。

・身近な日常生活のなかからでも感動を見つけることができるような，感性を身につけなければいけない。

このように考えると，作品を制作するという行為は，結局自分自身の人としての内容を高めるという行為にほかならないことに気づくでしょう。そして，その作品を多くの方に見てもらうことで，その成果を分かち合うのです。

“CGAを作ってどうするの？”と聞かれたとき，胸を張って答えたいものです。

さて来月は，CGAコンテスト以来，この連載が乱れがちだと反省しておりますので，立て直しの意味で久しぶりにお休みをいただきたいと思います。実は，単なる夏休みだったりして。それでは，また。

アマチュアCGA学会(その4)

タイトル：3D電子遊戯が人体に与える影響を軽減させるための考察
テストプレイヤー(以下TP)：かまた ゆたか
服用薬：市販の「乗り物の酔い止め薬」
電子遊戯：SUPER32X用「DOOM」

はじめに

最近CPUの高速化や新世代ゲーム機の出現によって，ポリゴンなどを使ったリアルな3D空間を自由に動き回れるような電子遊戯が多々発表された。しかし，一部の人間には，この種の電子遊戯によって，“見ているだけで気分が悪くなる”という現象が現れ，大きな弊害となっている。そこで，特にこの種の現象が著しく現れるTPに対して，特定の薬を服用することで，この影響を軽減することが可能であるかを調べることが，本研究の目的である。

なお，使用する薬については，同品に記載されている注意事項を十分確認のうえ，服用すること。

今回実験に使用した「DOOM」

実験方法

1）TPに，上記電子遊戯を実行させる。
2）気分が悪くなり，継続が不可能となるまでの時間を測定する。
3）また別の日，上記の薬を服用する。
4）1時間経過した後，1）と同様の条件で電子遊戯を行わせる。
5）2)と同様に時間を測定する。
6）上記の実験を数日ごとに繰り返す。
7）2)と5)の時間の平均を取り，その差を調べる。

上記電子遊戯は，3Dで描かれた基地の中を歩き回って敵を撃つといった内容であるが，単に3D空間を動くだけでなく，1歩進むごとに，上下に揺れるため，本実験に最適な遊戯であると思われる。

実験結果

・1日目
薬の服用なし　：約10分
・2日目
薬の服用あり　：約15分
・以下のデータなし

考察

上記の実験結果のように，薬を服用することで，電子遊戯による影響を約1.5倍軽減することができた。しかしながら，気分が悪くなること自体は回避できず，15分では遊戯をクリアすることができないため，根本的な解決策とはならないことがわかった。

なお，本実験は本来複数回繰り返し，その平均を得るべきであるが，TPが“オェッ”といって，実験を拒んだため，3回目以降の実行には至らなかった。

(で)のショートプロぱーてぃ――その72

「自分で作れ」の精神を見た！

Komura Satoshi 古村 聡

DIYの精神を実生活で実感してしまった(で)氏。貼り替え用の壁紙や透水タイル，そして電動ドリルなど，世の中便利なツールがあるんだなあ，と感心したからかわかりませんが，今月はちょっと便利なツールが3本です。

illustration:T.Takahashi

いやあ，「なければ自分で作れ！」なんていまどきいうのは，Oh!Xだけかと思っていたんですけど，やってるところはやってるんですね。部屋を探したんですけど，いわゆる貧乏ライターの悲しい性「予算の制約」っていうヤツ。なるべく安く部屋を探したところ，くそがつくほどのチョチョ，超ボロ部屋！　壁クロスビリビリ，シールの跡ペタペタ，柱に釘の穴空いてるし。なんとかしろよ，ごうつく大家！

さすがにこれはヤダ，でも金はない。しかたなく，いま風にいうとDIY，いわゆる日曜大工の店に行ってみたんです。場所が郊外だもんで，結構近くに体育館みたいなバカがつくようなでかいDIY屋がありました。そうしたら，あるわあるわ。貼り替え用の白亜でビビッド，マンションのモデルルームで使えそうな壁紙(しかも，最近の壁紙はシールになってて裏の台紙はがすだけで貼れるようになってるんですぜ！)，玄関に貼る色とりどりのおしゃれな透水タイル(これもシール)，ただの板を高級感あふれるチークっぽい色(あくまでも「っぽい」色)にする色つきニススプレー……などなど。なければ作れの精神はこんなところに生きていたんですね。俺らの知らんところでやってたなオヤヂども。あんまり感動したんで，私，思わず電気ドリルなんか1本衝動買いしてしまいましたもの。国産一流メーカー製4,480円，アタッチメントを替えると塗装はがし用金属ブラシや車磨き用のスポンジもつけられます。

とりあえず台所でも磨いてみるかな。うりゃあ！　このうなるモーター音がたまらんわ(うっとり)。

SX-BASICがゴニョゴニョなのだ

さて，DIY精神の重要さ感じつつ，今月の1本目はSX-BASIC上で動くユーティリティプログラムです。SND_Viewer.SXBです。どうぞっ！

SND_Viewer.SXB for X680x0
(要SX-BASIC)
群馬県　豊島隆志

「(で)さん，こんにちは。『ショートプロぱーてぃ』には初めての投稿となります。(「LIVE in '95」のコーナーには1回だけ投稿しました)。今回は(といっても初めてなのですが)SX-BASICでプログラムを作ってみました。FM音源を使う際の支援ツールです」

ということで投稿原稿そのまんま引用してしまう，悪い私(や～，楽だわこれ(笑))。このプログラムは，SNDファイルを自力で解析して作っちゃった力作なんであります よ。簡単にいえば「SOUND PRO-68K」(SX版のSOUND PROでも平気なはずです)で使う，SND形式ファイルの中身を表示するツールなんです。また，表示したデータをZMSの音色設定の形式にしてクリップボードにコピー，またはファイルとして作ることができるというすぐれもの。これによりFM音源の音色をSOUND PRO(SX)-68KでエディットしてこのツールによってZMS用データに変換する，ということができちゃうんですね。

なお，このプログラムは付録ディスク掲載バージョンのSX-BASIC ver.0.6以降用に作られていますので注意してね。SX-BASICを起動してリスト1を打ち込んでいってください。runすると表示するZMSファイルの名前を聞いてきますので，キーボードで入力してください。省略すると“a:\quickstart\sound.snd”になります。ファイルが読み込めたらあとはアップダウンボタンで音色番号を変更してください。その番号のデータが表示されます(10MHzや16MHzだと処理が重いんだけど，<DISP>ボタンをOFFにすればウィンドウの下半分が消えて，ちょっとだけ表示が速くなるので試してみてね)。

で，<コピー>ボタンを押すと，表示されているデータが.ZMSファイル用のデータに変換されて，クリップボードにコピーすると同時にそれと同内容のものを環境変数「temp」の示す場所に「SND_Viewer.ZMS」のファイル名で書き込んでくれます。

ただし，このバージョンのSX-BASICではバグのためにクリップボードにコピーしたデータをほかのアプリにペーストできません。ちょっと残念ですね。

まったくです！　SX-BASICを置いてどこへトンズラしちゃったんでしょ石上くんてば，ぷんぷん。これができればすごくおいしいアプリだったのに。ファイルの読み込みをすればコピーペーストと同じ結果を得ることはできますが，まったくもってまったくもって。ウィンドウ型のOSのいいところって，この窓から窓へのデータが簡単にやり取りできることなんですよね。この機能が使えればずいぶん便利になったのになぁ。おーい，石上くん，カムバーック！

ふ～む，でも石上くんがいないからといってSX-BASICを捨てることはありませんけどね。せっかく，SX-WINDOWでもお手軽プログラミングの可能性が見えてきて，すべてはこれから始まるのですから。深刻な後継者問題，なんていうと過疎化の進む農村みたいだけど，どなたかいい人いらっしゃらないかしらん。

そうそう，豊島さんが解析したSNDファイルの構造を掲載しておきます。

「まず，SNDファイルは1つの音色に対して，80(&H50)バイトの大きさをもちます。これが1番から200番まで順番に並んだ16000バイトのデータがSNDファイルです。各音色データのフォーマットは表のとおりです。独自に解析したので間違っているところもあるかもしれません」

とのことです。ユーティリティなんかを作るときには参考にしてくださいね。

ゴッソリ消しちゃうDELALL

続いて2本目のプログラムは，一発ポンできれいさっぱりなツール(?)，東京都の久我祐司さんのプログラムでDELALL.Xです。どうぞっ。

DELALL.X for X680x0

(要Cコンパイラ)

東京都　久我祐司

このプログラムは，指定したディレクトリ以下のファイルをすべて消すプログラムです。

リスト2を打ち込んで，

```
A>CC /Y /W /O /Gs30000 DELALL.C
```

で実行ファイルを作ってください。このプログラムでは，再帰を使っているので/Gsオプションでスタックを多めにとっています。コンパイルできましたか？　無事，実行ファイル，DELALL.Xができたら使い方です。

```
A>DELALL ディレクトリ名
```

でそのディレクトリ以下，ファイルもサブディレクトリもすべて消します。たとえば，

```
\TEMP + FILE1.$$$
      |
      + DIR1    - FILE2.$$$
```

というディレクトリがあったときに，

```
A>DELALL \TEMP
```

とすると\¥TEMP以下すべてを消します。

ただし，システムファイルや隠し属性のファイルがあった場合はそのディレクトリは消せません(消せるものだけを消しにいきますが)。COPYALLコマンドなどでHUMAN.SYSをコピーしたものがそれにあたります。注意してください。

ということで，ディレクトリ版のDELコマンドです。便利は便利なんだけど，実はこのプログラムの最初のバージョンでは，削除ファイルを確認せず，さっさとファイル消去をしてしまうものでした。やはり，それじゃあ安心して使えないということで，消されるファイル名一覧を表示して，なおかつ確認のメッセージを出すようにしました(まあ，削除ファイル名が表示されてる間に，ドキドキ感を味わえるのはいいんだけどね)。確認のメッセージを出すとはいえ，間違えてファイルを消しちゃうとファイルの復活ができませんから，プログラムを使うときには気をつけてくださいね。便利な道具は危険なものなのよね。ノミとかナタとかもそうですもんね。あと，電動ドリルとか。うふふ。

DIYに便利なusng(　)なのだ

さて，今月の3本目は，めずらしくX-BASICの「サブルーチンだけ」のプログラムなんであります。floatの数値を，指定の書式の文字列に変換してくれる，USNG.BASです。どうぞっ。

表　SNDファイルのフォーマット

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0			音色名(10文字)								AR				DR	
10			SR				RR				DL				TL	
20			KS				MT				DT1				DT	
30	2		AME				00	AF	WV	SP	PS	AS	PD	AD	00	00
40	SN	SM	PN	00	00	00	00	00	00	00	00	00	00	00	00	00

注)
- 音色名はコードそのままです。
- AL～AMEは4バイトありますが1バイト目がオペレータ1に，2バイト目がオペレータ2に……と対応しています。
- 「00」のところはおそらく未使用部分なので0で埋めます。
- 各文字は以下のような意味をもちます。

AR:Attack Rate
DR:First Decay Rate
SR:Second Decay Rate
RR:Release Rate
DL:First Decay Level
TL:Total Level
KS:Key Scaling
MT:Phase Multiply
DT1:Detune 1
DT2:Detune 2
AME:AME Enable
AF:フィードバック／アルゴリズム
WV:ウェーブフォーム
SP:スピード
PS:Pitch Modulation Sensitivity
AS:Amplitude Modulation Sensitivity
PD:Pitch Modulation Depth
AD:Amplitude Modulation Depth
SN:シンクロ
SM:スロットマスク
PN:パン

USNG.BAS for X680x0

(要X-BASIC)

福島県　熊田秀幸

え～，このプログラム，リスト3に掲載されていますが，単体で動くプログラムではありません。リスト3ではサンプルプログラムと一緒に掲載されています。このリスト3を打ち込んで，runしてみてちゃんと動いていることを確認したら，650行から先をSAVE@を使いセーブしてください。

では，このusng()関数の使い方です。このサブルーチンは浮動小数点の数値を，指定の書式に合わせて文字列に変換するサブルーチンです。プログラムのメイン部分にグローバル文字変数，

```
str usngstr
```

を定義しておいて(サンプルだと40行ですね)，このusng()を呼び出してください。呼び出しの形式は，

```
usng(float変数,"書式")
```

です。すると，指定した書式に合わせた形でfloat型変数が文字列に変換されて，usngstrに代入されます。

このプログラムではいろいろな書式を指定することができます。

```
"###.##"
```

以上のように指定すると#の個数で桁数を指定できます。この場合は小数点以上有効3桁，小数点以下2桁です。この書式でfloat型変数に12.345を入れて変換すると，

```
###.##
 12.35
```

となります(実は書式の「#」はこの文字でなくても，アルファベットでも数字でもスペースでもいんですが，X-BASICのusing文と同じように「#」で指定したほうがわかりやすいのでここでは「#」ということにしておきます)。

なお，書式を指定しない場合，つまり文字列に""を指定すると，システムの浮動小数点表記でusngstrに代入されます。

```
usng(sqr(3),"")
usngstr→1.7320508075689
```

それから，「#」と一緒に，特別な文字を入れておくことでいろいろな変換をすることもできます。

```
USNG(255,"H##.##")
```

以上のようにするとusngstrには"FF"が入ります。つまり16進数に変換してくれるのですね。さらに0(ゼロ)を付け加えると，先頭からの空白を「0(ゼロ)」にすることができます。

```
usng(255,"0h#.##")
usngstr → 0FF
```

アルファベットのO(オー)は8進数への変換です。またこのオプションにも0指定が可能です。

usng(255,"O###.##")

usngstr → 337

usng(255,"0o##.##")

usngstr → 0337

そして，アルファベットのBは2進数に変換。これも0指定ができます。

usng(255,"B########.##")

usngstr → 11111111

usng(255,"0B########.##)

usngstr → 0011111111

ということで，16/8/2進指定のできるusingなのでありますね。そうそう，2進で表示しようとか思っていて，いつも一番悩んじゃうのがこの「表示するのに必ず16桁にして，あまった左の桁は0を表示するようにしたい……でもめんどくさい」なんでありますよね！　でしょでしょ。そんなときにこの1本なんであります。

きっと作者の熊田さんも「めんどくさいから使い回しできるようにしてしまえ！」ってことで作ってしまったのでしょう。おかげさまでシールで貼れる玄関タイルみたいに，私たちにも使いやすくなってます。ほら，タイル貼るのってセメント練ったり砂を買ってきてタイルの間につめたり大変なんですよ。このプログラムは，作ってしまえの人が，作りたい人を助ける助け合いプログラムですね。熊田さんに感謝感謝(どうでもいいけど日曜大工屋に砂を使わず水を混ぜるだけ，3時間で乾くインスタントセメントってのがあったな)。

さて，どうでもいいけど，「もしかして壁がキレイになるかな。引っ越す前に磨いておくのもいいかな」と思って，ブラシをつけて電気ドリルで旧居の壁を磨いてみたら(ちょっと使ってみたくて……我慢できなかったんだよ)。や～，キレイに丸く壁のペンキがハゲてしまった。

ま，失敗も成功の母ってことで，あなたは恐れずに今月のプログラムを使ってみてくださいね(?)。プログラムじゃ壁はハゲませんから。

リスト1　SND_Viewer.SXB

```
     1: ▼Window Size (316,220),0,1,1,SND File Viewer びーばー査運奴君
Ver0.99 + α
     2: str filename,voicename
     3: dim char data(16000)
     4: int voicenumber=0
     5: int Display=1
     6: int i,j,file
     7: filename=inputbox$("表示するファイル名を入力して下さい")
     8: if filename="" then filename="a:¥quickstart¥sound.snd"
     9: File_Drop(filename)
    10: AlterBtn1.value = 1
    11:
    12: func File_Drop(fn;str)
    13:  if right$(fn,3)<>"snd" and right$(fn,3)<>"SND" then return(-1)
    14:  Text2.caption = " File:"+fn
    15:  filename=fn
    16:  file=fopen(filename,"r")
    17:  if file<>-1 then {
    18:      di()
    19:         fread(data,16000,file)
    20:         fclose(file)
    21:      ei()
    22:  }
    23:  Put_Data()
    24: endfunc
    25:
    26: ▼7,UpdwnBtn1 (8,33,80,52),0,0,0,199,0,1,   0
    27: func UpdwnBtn1_Change(val;int)
    28:    UpdwnBtn1.enable = 0
    29:    voicenumber=val
    30:    Put_Data()
    31:    UpdwnBtn1.enable = 1
    32:    AlterBtn1.value = Display
    33: endfunc
    34:
    35: ▼2,Rect1 (8,60,308,212),0,0,1
    36:
    37: ▼1,Text1 (84,28,308,52),0,0,1,0,3,0,0,1,音色名:
    38: func Text1_Click()
    39: endfunc
    40:
    41: ▼1,Text2 (96,6,308,24),0,0,0,0,3,0,0,1, File:
    42: func Text2_Click()
    43: endfunc
    44:
    45: ▼1,Text3 (60,2,88,14),0,0,0,1,3,0,0,0,DISP
    46: func Text3_Click()
    47: endfunc
    48:
    49: ▼1,Text4 (48,72,300,88),0,0,0,1,7,0,0,0,AR  DR  SR  RR  SL  OL
KS  ML DT1 DT2 AME
    50: func Text4_Click()
    51: endfunc
    52:
    53: ▼1,Text5 (16,88,36,104),0,0,0,1,7,0,0,0,Op1
    54: func Text5_Click()
    55: endfunc
    56:
    57: ▼1,Text6 (16,108,36,124),0,0,0,1,7,0,0,0,Op2
    58: func Text6_Click()
    59: endfunc
    60:
    61: ▼1,Text7 (16,128,36,144),0,0,0,1,7,0,0,0,Op3
    62: func Text7_Click()
    63: endfunc
    64:
    65: ▼1,Text8 (16,148,36,164),0,0,0,1,7,0,0,0,Op4
    66: func Text8_Click()
    67: endfunc
    68:
    69: ▼1,Text9 (48,168,108,184),0,0,0,1,5,0,0,0,AL  FB  OM
    70: func Text9_Click()
    71: endfunc
    72:
    73: ▼1,Text10 (16,188,40,204),0,0,0,1,5,0,0,0,Etc.
    74: func Text10_Click()
    75: endfunc
    76:
    77: ▼3,StnBtn1 (8,4,48,24),0,0,コピー
    78: func StnBtn1_Click()
    79: str rem1="/      AR  DR  SR  RR  SL  OL  "
    80: str rem2="KS  ML DT1 DT2 AME    "
    81: str rem3="/      AL  FB  OM"
    82: str ret=chr$(&h0d)+chr$(&h0a)
    83: dim str op(8)={"","       ","       ","       ","","","","","
"}
    84: int no=voicenumber*&h50+&ha
    85:    UpdwnBtn1.enable = 0
    86:    di()
    87:    op(0)="(@"+right$("  "+str$(voicenumber+1),3)+","
    88:    for i=0 to 3
    89:     for j=0 to 5
    90:          op(i)=op(i)+right$("  "+str$(data(no+j*4+i)),3)+","
    91:     next
    92:     for j=6 to 10
    93:      op(i+4)=op(i+4)+right$("  "+str$(data(no+j*4+i)),3)
    94:      if j=10 then op(i+4)=op(i+4)+ret else op(i+4)=op(i+4)+","
    95:     next
    96:    next
    97:       op(8)="        "+right$("  "+str$(data(no+&h2d) mod 16),3)+"
,"+right$("  "+str$(data(no+&h2d)¥16),3)+","+right$("  "+str$(data(no+&
h37)),3)+")"+ret
    98:       clipboard.strn = rem1+rem2+voicename+ret+op(0)+op(4)+op(1)
+op(5)+op(2)+op(6)+op(3)+op(7)+rem3+ret+op(8)
    99:    ei()
   100:    Data_Save()
   101:    UpdwnBtn1.enable = 1
   102: endfunc
   103:
   104: ▼5,AlterBtn1 (56,16,88,28),0,0,1
   105: func AlterBtn1_Change(val;int)
   106:    UpdwnBtn1.enable = 0
   107:    if val=0 then {
   108:       window.height = 58
   109:    }
   110:    if val=1 then {
   111:       Put_OpData()
   112:       window.height = 220
   113:    }
   114:    Display=val
   115:    UpdwnBtn1.enable = 1
   116:    AlterBtn1.value = Display
   117: endfunc
   118:
   119: ▼1,Text (40,88,60,104),1,0,0,0,3,0,-1,1,
   120: ▼1,Text (40,108,60,124),1,1,0,0,3,0,-1,1,
   121: ▼1,Text (40,128,60,144),1,2,0,0,3,0,-1,1,
   122: ▼1,Text (40,148,60,164),1,3,0,0,3,0,-1,1,
   123: ▼1,Text (64,88,84,104),1,4,0,0,3,0,-1,1,
   124: ▼1,Text (64,108,84,124),1,5,0,0,3,0,-1,1,
   125: ▼1,Text (64,128,84,144),1,6,0,0,3,0,-1,1,
   126: ▼1,Text (64,148,84,164),1,7,0,0,3,0,-1,1,
   127: ▼1,Text (88,88,108,104),1,8,0,0,3,0,-1,1,
   128: ▼1,Text (88,108,108,124),1,9,0,0,3,0,-1,1,
   129: ▼1,Text (88,128,108,144),1,10,0,0,3,0,-1,1,
   130: ▼1,Text (88,148,108,164),1,11,0,0,3,0,-1,1,
   131: ▼1,Text (112,88,132,104),1,12,0,0,3,0,-1,1,
   132: ▼1,Text (112,108,132,124),1,13,0,0,3,0,-1,1,
   133: ▼1,Text (112,128,132,144),1,14,0,0,3,0,-1,1,
   134: ▼1,Text (112,148,132,164),1,15,0,0,3,0,-1,1,
   135: ▼1,Text (136,88,156,104),1,16,0,0,3,0,-1,1,
   136: ▼1,Text (136,108,156,124),1,17,0,0,3,0,-1,1,
   137: ▼1,Text (136,128,156,144),1,18,0,0,3,0,-1,1,
   138: ▼1,Text (136,148,156,164),1,19,0,0,3,0,-1,1,
   139: ▼1,Text (160,88,180,104),1,20,0,0,3,0,-1,1,
   140: ▼1,Text (160,108,180,124),1,21,0,0,3,0,-1,1,
   141: ▼1,Text (160,128,180,144),1,22,0,0,3,0,-1,1,
   142: ▼1,Text (160,148,180,164),1,23,0,0,3,0,-1,1,
   143: ▼1,Text (184,88,204,104),1,24,0,0,3,0,-1,1,
   144: ▼1,Text (184,108,204,124),1,25,0,0,3,0,-1,1,
   145: ▼1,Text (184,128,204,144),1,26,0,0,3,0,-1,1,
   146: ▼1,Text (184,148,204,164),1,27,0,0,3,0,-1,1,
   147: ▼1,Text (208,88,228,104),1,28,0,0,3,0,-1,1,
   148: ▼1,Text (208,108,228,124),1,29,0,0,3,0,-1,1,
   149: ▼1,Text (208,128,228,144),1,30,0,0,3,0,-1,1,
   150: ▼1,Text (208,148,228,164),1,31,0,0,3,0,-1,1,
   151: ▼1,Text (232,88,252,104),1,32,0,0,3,0,-1,1,
   152: ▼1,Text (232,108,252,124),1,33,0,0,3,0,-1,1,
   153: ▼1,Text (232,128,252,144),1,34,0,0,3,0,-1,1,
   154: ▼1,Text (232,148,252,164),1,35,0,0,3,0,-1,1,
   155: ▼1,Text (256,88,276,104),1,36,0,0,3,0,-1,1,
   156: ▼1,Text (256,108,276,124),1,37,0,0,3,0,-1,1,
   157: ▼1,Text (256,128,276,144),1,38,0,0,3,0,-1,1,
   158: ▼1,Text (256,148,276,164),1,39,0,0,3,0,-1,1,
   159: ▼1,Text (280,88,300,104),1,40,0,0,3,0,-1,1,
```

```
160:  ▼1,Text (280,108,300,124),1,41,0,0,3,0,-1,1,
161:  ▼1,Text (280,128,300,144),1,42,0,0,3,0,-1,1,
162:  ▼1,Text (280,148,300,164),1,43,0,0,3,0,-1,1,
163:  ▼1,Text (40,184,60,200),1,44,0,0,3,0,-1,1,
164:  ▼1,Text (64,184,84,200),1,45,0,0,3,0,-1,1,
165:  ▼1,Text (88,184,108,200),1,46,0,0,3,0,-1,1,
166: func Text_Click(index;int)
167: endfunc
168:
169:  ▼1,Text11 (144,188,300,204),0,0,0,1,2,0,1,0,Programed by T.Toyo
shima
170: func Text11_Click()
171: endfunc
172:
173: /******
174:
175: func Put_Data()
176:    di()
177:       voicename=""
178:       for i=0 to 9
179:          voicename=voicename+chr$(data(voicenumber*&h50+i))
180:       next
181:    ei()
182:    Text1.caption = "音色名："+voicename
183:    UpdwnBtn1.caption = right$("   "+str$(voicenumber+1),4)
184:    if Display=1 then Put_OpData()
185: endfunc
186:
187: func Put_OpData()
188: int no=voicenumber*&h50+&ha
189:  di()
190:   for i=0 to 43
191:     Text[i].caption = str$(data(no+i))
192:   next
193:   Text[44].caption = right$("  "+str$(data(no+&h2d) mod 16),3)
194:   Text[45].caption = right$("   "+str$(data(no+&h2d)¥16),3)
195:   Text[46].caption = right$("  "+str$(data(no+&h37)),3)
196:  ei()
197: endfunc
198:
199: /*--------------------
200:
201: func Data_Save()
202: str savename
203: int fn
204: str rem1="/      AR  DR  SR  RR  SL  OL  "
205: str rem2="KS  ML DT1 DT2 AME   "
206: str rem3="/      AL  FB  OM"
207: str ret=chr$(&h0d)+chr$(&h0a)
208: dim str op(8)=("","      ","      ","      ","","","","
")
209: int no=voicenumber*&h50+&ha
210:    di()
211:    op(0)="(@"+right$("  "+str$(voicenumber+1),3)+","
212:    for i=0 to 3
213:       for j=0 to 5
214:       op(i)=op(i)+right$("  "+str$(data(no+j*4+i)),3)+","
215:       next
216:       for j=6 to 10
217:       op(i+4)=op(i+4)+right$("  "+str$(data(no+j*4+i)),3)
218:       if j=10 then op(i+4)=op(i+4)+ret else op(i+4)=op(i+4)+","
219:       next
220:    next
221:    op(8)="      "+right$("  "+str$(data(no+&h2d) mod 16),3)+","+
right$("  "+str$(data(no+&h2d)¥16),3)+","+right$("  "+str$(data(no+&h37
)),3)+")"+ret
222:    savename=getenv("temp",0)+"SND_Viewer.ZMS"
223:    fn=fopen(savename,"c")
224:    fwrites(rem1+rem2+voicename+ret+op(0)+op(4)+op(1)+op(5)+op(2)
+op(6)+op(3)+op(7)+rem3+ret+op(8),fn)
225:    fclose(fn)
226:    ei()
227: endfunc
```

リスト2　DELALL.X

```
  1: #include <stdio.h>
  2: #include <doslib.h>
  3: #include <error.h>
  4: #include <string.h>
  5: #include <direct.h>
  6: #define _DISPLAY 0
  7: #define _DELETE 1
  8:
  9: int i;
 10:
 11: static char *HelpMes[]={
 12:   "ディレクトリ消去プログラム  DELDIR.X    Ver1.00",
 13:   "                            (c)Dejavu soft 1995",
 14:   "[使用方法]                                      ",
 15:   "      A>DELDIR ディレクトリ名                ",
 16:   "      ディレクトリ以下の内容をすべて消します"};
 17:
 18: void help(void)
 19: {
 20:    printf("¥n");
 21:    for(i=2;i<5;i++)
 22:       printf("%s¥n",HelpMes[i]);
 23: }
 24:
 25: void DeleteAll(char *path,int isDel){
 26:    struct FILBUF c_file;
 27:    char szFilename[256];
 28:
 29:    /*ディレクトリにある拡張子が*.*の最初のファイルを検索 */
 30:    if(isDel==_DISPLAY)
 31:    printf("--[%s]--------------------------------¥n",path);
 32:    strcpy(szFilename,path);
 33:    strcat(szFilename,"¥¥*.*");
 34:    FILES( &c_file, szFilename, 0x30 );
 35:    if((strcmp(c_file.name,"..") != 0)&&(strcmp(c_file.name,".")
!= 0)){
 36:       if((c_file.atr && 0x20) == 0x20){
 37:
 38:          /*ディレクトリだった*/
 39:             char NewDir[256];
 40:
 41:             strcpy(NewDir,path);
 42:             strcat(NewDir,"¥¥");
 43:             strcat(NewDir,c_file.name);
 44:             DeleteAll(NewDir,isDel);
 45:             if(isDel == _DELETE )
 46:                rmdir(NewDir);
 47:             else
 48:                printf(">RD %s¥n",NewDir);
 49:       }else{
 50:
 51:          /*ディレクトリでない*/
 52:          char Filename[256];
 53:          strcpy(Filename,path);
 54:          strcat(Filename,"¥¥");
 55:          strcat(Filename,c_file.name);
 56:          if(isDel == _DISPLAY )
 57:             printf(">DEL %s¥n",Filename);
 58:          else
 59:             remove(Filename);
 60:       }
 61:    }
 62:    /*                                  */
 63:    /* 拡張子が*.*の残りのファイルを検索 */
 64:    /*                                  */
 65:    while( NFILES( &c_file ) == 0 ){
 66:        if((strcmp(c_file.name,"..") != 0)&&(strcmp(c_file.name,"
.") != 0)){
 67:           if((c_file.atr & 0x10) == 0x10){
 68:
 69:              /*ディレクトリだった*/
 70:              char NewDir[256];
 71:
 72:              strcpy(NewDir,path);
 73:              strcat(NewDir,"¥¥");
 74:              strcat(NewDir,c_file.name);
 75:              DeleteAll(NewDir,isDel);
 76:              if( isDel ==_DELETE )
 77:                 rmdir(NewDir);
 78:              else
 79:                  printf(">RD %s¥n",NewDir);
 80:           }else{
 81:
 82:              /*ディレクトリでない*/
 83:              char Filename[256];
 84:              strcpy(Filename,path);
 85:              strcat(Filename,"¥¥");
 86:              strcat(Filename,c_file.name);
 87:              if(isDel == _DISPLAY)
 88:                 printf(">DEL %s¥n",Filename);
 89:              else
 90:                 remove(Filename);
 91:           }
 92:       }
 93:    }
 94:    return;
 95: }
 96:
 97: main(int argc,char* argv[])
 98: {
 99:    int ch;
100:
101:    printf("¥n");
102:    for(i=0;i<2;i++)
103:       printf("%s¥n",HelpMes[i]);
104:
105:    if(argc != 2) help();
106:    else{
107:       printf("¥n");DeleteAll(argv[1],_DISPLAY);
108:       printf(">RD %s¥n",argv[1]);
109:       printf("以上を削除します。よろしいですか？[Y]/n");
110:       ch = getche();printf("¥n");
111:
112:       if((ch == 'Y')||(ch=='y')||(ch=='¥n')){
113:        DeleteAll(argv[1],_DELETE);
114:        if(0 == rmdir(argv[1])){
115:         printf("¥nディレクトリ%sは削除されました.¥n",argv[1]);
116:        }else printf("%sは削除できません.¥n",argv[1]);
117:       }
118:
119:    }
120:    return 0;
121: }
```

リスト3　USNG.BAS

```
 10 float g,h,i
 20 str l,m
 30 /****************
 40 str usngstr
 50 /****************
 60 h=255.234#
 70 usng(h,"")
 80 l="             "
 90 print h,l,usngstr
100 print
110 /*
120 l="##########.##"
130 usng(h,l)
140 print h,l,usngstr
150 print
160 /*
```

▶「ANOTHER CG WORLD」，終わっちゃうんですね。江口響子さんの絵は大好きなのでとても残念です。「また近いうちに……」という言葉に期待しています。
岡田　千穂(22)神奈川県

```
170 l="#0########.##"
180 usng(h,l)
190 print h,l,usngstr
200 print
210 /*
220 h=-255.234#
230 l="##########.##"
240 usng(h,l)
250 print h,l,usngstr
260 print
270 /*
280 h=-255.234#
290 l="#0########.##"
300 usng(h,l)
310 print h,l,usngstr
320 print
330 /*
340 h=255.234#
350 l="h#########.##"
360 usng(h,l)
370 print h,l,usngstr
380 print
390 /*
400 h=255.234#
410 l="H0########"
420 usng(h,l)
430 print h,l,usngstr
440 print
450 /*
460 l="o#########.##"
470 usng(h,l)
480 print h,l,usngstr
490 print
500 /*
510 l="O0########"
520 usng(h,l)
530 print h,l,usngstr
540 print
550 /*
560 l="b#########.##"
570 usng(h,l)
580 print h,l,usngstr
590 print
600 /*
610 l="B0########"
620 usng(h,l)
630 print h,l,usngstr
640 end
650 /*****************************************************************
660 func usng(a;float,fmt;str)
670 /***  浮動小数点の数値を指定の書式の文字列にする。ﾒｲﾝﾙｰﾁﾝ ***
680 /*****************************************************************
690 float b,c,d
700 int col,co2,l
710 /*
720         if a=0# then {
730                 us0(" ",fmt)
740         } else {
750                 col=instr(1,fmt,".")
760                 d=abs(a)
770                 l=len(fmt)
780                 if col=0 then {
790                         if (d>=1) or (l=0) then col=1
800                 } else {
810                         co2=l-col
820                         c=co2
830                         b=1#/pow(10#,c)
840                         if d>b then col=1 else col=0
850                 }
860                 if col=0 then {  /*** 小数点
870                         if a<0 then us0("-",fmt)
880                         if a>0 then us0("+",fmt)
890                 } else {
900                         usl(a,fmt)
910                 }
920         }
930 /*
940 endfunc
950 /*******************************************************************
960 func us0(d;str,l;str):/*** 「０」を書式指定の文字列にする。***
970 /*******************************************************************
980 int a,b,c,g,i
990 str f,h,e,sp
1000 /***
1010 h=""
1020 a=len(l)
1030 b=instr(1,l,".")
1040 i=instr(1,l,"0")
1050 if i=0 then {
1060         sp=" "
1070 } else {
1080         sp="0"
1090         d="0"
1100 }
1110 if (a=1) or (b=2) then d=""
1120 if b<>0 then {
1130         h="."
1140         c=a-b
1150         if c>0 then h=h+string$(c,"0")
1160         b=b-1
1170 } else {
1180         b=a
1190 }
1200 f=""
1210 e=d+"0"
1220 g=len(e)
1230 repeat
1240         f=f+mid$(e,g,1)
1250         g=g-1
1260         b=b-1
1270 until g=0
1280 /*
1290 while b>0
1300         f=f+sp
1310         b=b-1
1320 endwhile
1330 f=mirror$(f)
1340 usngstr=f+h
1350 /***
1360 endfunc
1370 /***************************************************************
1380 func usl(i;float,l;str)
1390 /*** 浮動小数点の数値を指定の書式に合わせた文字列にする。***
1400 /***************************************************************
1410 int a,b,c,d,e
1420 float j
1430 str m,n,o,p,s,sp
1440 int q,r
1450 int tv=0,t0,t
1460 /**
1470 if l<>"" then {  /*  書式がある場合
1480 /**
1490         switch 1
1500         default
1510                 /*** １６進数
1520                 tv=instr(1,l,"h")
1530                         if tv<>0 then {
1540                                 tv=1
1550                                 break
1560                         }
1570                 tv=instr(1,l,"H")
1580                         if tv<>0 then {
1590                                 tv=1
1600                                 break
1610                         }
1620                 /***   ８進数
1630                 tv=instr(1,l,"o")
1640                         if tv<>0 then {
1650                                 tv=2
1660                                 break
1670                         }
1680                 tv=instr(1,l,"O")
1690                         if tv<>0 then {
1700                                 tv=2
1710                                 break
1720                         }
1730                 /***   ２進数
1740                 tv=instr(1,l,"b")
1750                         if tv<>0 then {
1760                                 tv=3
1770                                 break
1780                         }
1790                 tv=instr(1,l,"B")
1800                         if tv<>0 then tv=3
1810         endswitch
1820 /*** 文字列の先頭に０を置く
1830         t0=instr(1,l,"0")
1840 /*** 文字列変換
1850         a=instr(1,l,".")
1860         if a=0 then {
1870                 a=len(l)
1880                 s=""
1890                 c=a
1900         } else {
1910                 s="."
1920                 c=a-1
1930         }
1940         e=len(l)
1950         j=e-a
1960 /*
1970         switch tv
1980            case 0 :p=fcvt(i,j,q,r)
1990                 if r=0 then {
2000                         if i<1 then {
2010                             p="0"+string$(abs(q),"0")+p
2020                             q=1
2030                         }
2040                 } else {
2050                         if i<0 then {
2060                             if i<-1 then {
2070                                 q=q+1
2080                                 p="-"+p
2090                             } else {
2100                                 p="-0"+string$(abs(q),"0")+p
2110                                  q=2
2120                             }
2130                         }
2140                 }
2150                 break
2160            case 1 :t=i
2170                         p=hex$( t )
2180                         q=len( p )
2190                         break
2200            case 2 :t=i
2210                         p=oct$( t )
2220                         q=len( p )
2230                         break
2240            case 3 :t=i
2250                         p=bin$( t )
2260                         q=len( p )
2270         endswitch
2280 /***  整数部
2290         d=q
2300         if t0=0 then sp=" " else sp="0"
2310         n=""
2320         repeat
2330                 n=n+mid$(p,d,1)
2340                 c=c-1
2350                 d=d-1
2360         until d=0
2370         if c>0 then {
2380                 while c>0
2390                         n=n+sp
2400                         c=c-1
2410                 endwhile
2420         }
2430         n=mirror$(n)
2440 /***  小数部
2450         if a<>0 then {
2460                  d=q+1
2470                  o=mid$(p,d,255)
2480         }
2490         m=n+s+o
2500 } else {
2510         m=str$( i )
2520 }
2530 /*
2540 usngstr=m
2550 /*
2560 endfunc
```

▶ ７月号の「STUDIO X」の上田君へ。入部初日にパソコンを分解した石川君にも驚いたが，初心者のうちからジャンク屋巡りをしている君にも結構びっくりしたぞ。

加藤　一樹(21)埼玉県

音声波形表示プログラム

OCR.X

Ueda Tsuyoshi 上田　剛

オーディオINからの音声をリアルタイムに画面表示するツールです。そのほか，音声をサンプリングしてFFT解析することもできます。できるだけ単純な音を入れないと意味がありませんので，念のため。

OCRは音声信号をリアルタイムに処理するデジタルオシロスコープ&FFTアナライザもどきのプログラムです。

これはオーディオIN端子から入力された信号をサンプリングして，リアルタイムで波形を表示，またFFTなどの処理をするプログラムです。AD PCMでのサンプリングなのであまり高い周波数の信号とか方形波，DC波などの信号は正しくデータとして出力されません。

また，ADコンバータの精度も8ビットなので，計測器としてはなんの役にも立ちませんが（X68000のPCMが素直な16ビットPCMだったらよかったのに），ディスプレイの飾りにでも使用してください。

OCR.X 使用法

起動は，ただ，

　OCR〈リターン〉

で特にスイッチなどのオプションはありません。

X68000のオーディオIN端子に通常のオーディオレベルの信号を接続します（ラジカセなどのLINE OUTまたはPHONEからのものでよい）。

起動すると，オシロスコープのようなコンソールパネルとスクリーンが表示されますのであとは，マウスで各ボタンをクリックし操作してください。

OCRの画面。下は逆FFTの結果

各ボタンの説明

●上面のコンソールの説明

いちばん上にある「MODE」スイッチは現在使用できません。その下にある「EXIT」はOCR.Xを終了させるものです。

このOCR.Xでは一部，割り込みベクタを書き換えているため，なんらかのエラーが発生した場合（たぶんインタラプトスイッチくらいだと思うけど），このベクタの復帰がなされないでHuman68kに戻る場合があります。

また，Z-MUSICなどのタイマ割り込みを使用しているソフトが常駐していると正常に動作しませんので，おかしいと思ったらリセットしてください（EXITで終了すれば大丈夫）。

「POSITION」は，スクリーンに表示されている波形の位置を変更するものです。上下左右のボタンを左クリックでゆっくり，右クリックで早く移動します。

●下面のコンソールの説明

DISP

クリックすることより入力された信号の波形をほぼリアルタイムで表示します。

なお，このスイッチ以外の場所で左クリックすると停止します。

Y_SCALE

入力信号の波高値の表示をそれぞれ「1/1,1/2,1/4,1/8,1/16」にします。

SAMPLE_[kHz]

サンプリング周波数の変更をします，トリガーの関係から15.6kHz以外では，波形の表示が多少カクカクします。

SAMPLE

入力信号を内部バッファに取り込みます。同時にその信号の周波数，信号レベル（最大，最小32767），実効値，正弦波と比較した波形率，その他を計算表示します。

MONITOR

音としてX68000のスピーカーから鳴らす場合に使用します。

GRID

スクリーンにグリッドを表示します。

FFT

「SAMPLE」で取り込んだ信号をFFTにかけて0～49次の周波数成分に分解しスクリーン下面にグラフ表示します。

DATA

FFTで計算したものを，グラフではなく数値データとして表示します。表示は複素数形式とパワーのみの表示が順次切り替わります。

CLR

スクリーン上をクリアします。

RECT

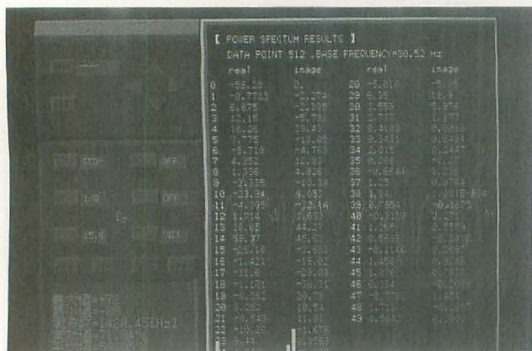
FFTにかけるデータに窓関数をかぶせます。方形波関数（そのまま），ハニング関数，ブラックマンハリス関数，ローゼンフィールド関数がクリックごとに切り替わります。

なお，FFTの計算結果の表示は15.6kHzでサンプリングされた場合を基準にしています。15.6kHzでの512ポイント分の1周波は，30.52HzとなりますのでFFTのグラフ上の1次分は30.52Hz，2次分は61.04Hz……となります。

＊　　＊　　＊

なお，参考としてADPCM→PCM変換の部分はZVT.Xから，リアルタイムでのPCM変換およびグラフ表示の方法は，かつてOh!Xに掲載されていたPCM8の解説と，『Inside X68000』（桒野雅彦著）よりヒントをいただきました。

それにしても，X68000のパワーユーザーといわれる人たちは，すごい人たちばかりですね。いつも，感謝とともに圧倒されています。

リスト1 OCR.C

```
  1: #include         "wave.h"
  2: 
  3: extern signed             short   c_pcmbuf[15600]; /*draw_wave.c より参照*/
  4: signed  short    pcmbuf[PCMSIZE];
  5: unsigned          char    vol_sift;
  6: unsigned          char    samp;
  7: unsigned          char    clk;
  8: unsigned          char    pan;
  9: int      data_cnt=0;
 10: int      wind_fnc=0;
 11: volatile          unsigned short  *gpalet;
 12: unsigned          short   *tpalet;
 13: 
 14: struct LINEPTR            lineptr;
 15: struct SYMBOLPTR          symbolptr;
 16: struct FILLPTR            fillptr;
 17: struct FILLPTR            *ptr_fill=&fillptr;
 18: struct LINEPTR            *ptr_line=&lineptr;
 19: struct SYMBOLPTR          *ptr_symbol=&symbolptr;
 20: static  unsigned          int     mask_y=0x0000ffff;
 21: static  unsigned          int     CRTMOD_ORG;
 22: 
 23: static  unsigned          short   new_gpalet[16]={0x0003,0x4211,0x4a55,0x83e3,0xce73,
 24:             0x3059,0x5261,0x6aa5,0x6339,
 25:             0x2b89,0x0e0f,0x17c3,
 26:             0x8e81,0xffc1,
 27:             0xfa01,
 28:             0xffff
 29: };
 30: 
 31: static unsigned char   mes1[]={"OCR Ver0.5  T.U  1994,95"};
 32: static unsigned char   mes2[]={"EXIT"};
 33: static unsigned char   mes3[]={"MODE"};
 34: static unsigned char   mes4[]={"----"};
 35: static unsigned char   mes5[]={"STLAGE"};
 36: static unsigned char   mes6[]={"DISP"};
 37: static unsigned char   mes7[]={"Y_SCALE"};
 38: static unsigned char   mes8[]={"SAMPLE_[kHz]"};
 39: static unsigned char   mes9[]={"MONITOR"};
 40: static unsigned char   mes10[]={"GRID"};
 41: static unsigned char   mes11[]={"RUN"};
 42: static unsigned char   mes12[]={"STOP"};
 43: static unsigned char   mes13[]={"▲"};
 44: static unsigned char   mes14[]={"POSITION"};
 45: static unsigned char   mes15[]={"SAMPL"};
 46: static unsigned char   mes16[]={"ON"};
 47: static unsigned char   mes17[]={"OFF"};
 48: static unsigned char   mes18[]={"FFT"};
 49: static unsigned char   mes19[]={"CLR"};
 50: static unsigned char   mes20[]={"DATA"};
 51: static unsigned char   mes_str[][6]={{"1/1"},{"1/2"},{"1/4"},{"1/8"},{"1/16"}};
 52: static unsigned char   mes_str2[][6]={{"3.9"},{"5.2"},{"7.8"},{"10.4"},{"15.6"}};
 53: static unsigned char   mes_str3[][6]={{"ｸﾛｽ"},{"ﾒｯｼｭ"},{"ｸﾘｱ"}};
 54: static unsigned char   mes_str4[][6]={{"左右"},{"右"},{"左"},{"OFF"}};
 55: static unsigned char   mes_str5[][6]={{"Rect"},{"Hann"},{"Blac"},{"Rose"}};
 56: static unsigned short  text_palet[16];
 57: 
 58: /*インターラプトスイッチで強制的に終了処理される*/
 59: /*break_key,copy_keyは無効にする*/
 60: volatile         void    (*func_vect)();
 61: static unsigned int     org1_vector;    /*normal dma interrupt  address*/
 62: static unsigned int     org2_vector;    /*normal interrupt  address*/
 63: static unsigned int     org3_vector;    /*HD copy*/
 64: static unsigned int     org4_vector;    /*break*/
 65: 
 66: static unsigned int     niv_no=0x6a;    /*dma interrupt bector nomber*/
 67: /*static unsigned        int     intrapt_sw=0x2e; エラー表示  trap E bector nomber*/
 68: static unsigned int     prn_break=0x2c; /*ハードコピーtrap C bector nomber*/
 69: static unsigned int     break_key=0x2b; /*break_key trap B bector nomber*/
 70: 
 71: 
 72: void    main(){
 73: 
 74:     int ms_x;
 75:     int ms_y;
 76:     int ms_temp;
 77:     int j=0,k=4,l=0,s=3;
 78:     int h_x=0;
 79:     int h_y=0;
 80: 
 81:     PCM_BANK  *pcmhead; /*(環状双方向リスト形式)のヘッド*/
 82:     PCM_BANK  *pcmbottm;        /*     〃       現在 ADR        */
 83:     FILE_LIST *file_head;
 84:     FILE_LIST *file_bottm;
 85: 
 86:     vol_sift=3;
 87:     samp=0x0b;
 88:     clk=0x0;
 89:     pan=0x3;
 90: 
 91:     if(TGUSEMD(0,-1)==1 || TGUSEMD(0,-1)==2){
 92:         printf("グラフィックが使用出来ません.¥n");
 93:         exit(1);
 94:     }
 95: 
 96:     pcmhead=pcmbottm=malloc(sizeof(PCM_BANK));
 97:     file_head=file_bottm=malloc(sizeof(FILE_LIST));
 98: 
 99:     if(pcmhead==NULL || file_head==NULL){/*この時点でメモリが無いと致命的*/
100:         fprintf(stderr,"メモリが有りません終了します!¥n");
101:         exit(1);
102:     }
103: 
104:     pcmhead->buf_name[0]=NULL;
105:     file_head->file_name[0]=NULL;
106: 
107:     func_vect=(void (*)())warikomi;
108:     org1_vector=INTVCS(niv_no,(char *)func_vect);        /*dma normalベクター処理先addressを
入れ替える*/
109: 
110: /*      func_vect=(void (*)())er_break;
111:     org2_vector=INTVCS(intrapt_sw,(char *)func_vect); trap Eベクター処理先addressを入れ替える
112: */
113:     func_vect=(void (*)())hd_copy;
114:     org3_vector=INTVCS(prn_break,(char *)func_vect);     /*trap Cベクター処理先addressを入れ替
える*/
115:     org4_vector=INTVCS(break_key,(char *)func_vect);     /*trap Bベクター処理先addressを入れ替
える*/
116: 
117:     init_screen();
118:     init2_console();
119:     VPAGE(15);
120: 
121: while(1){
122:         /*左ボタン系*/
123:     if(MS_GETDT()==0xff00){
124:         ms_temp=MS_CURGT();
125:         ms_y=mask_y & ms_temp;
126:         ms_x=ms_temp>>16;
127: 
128:         if(ms_x>65 && ms_x<95){
129:             if(ms_y>320 && ms_y<340){   /*dataボタン*/
130:                 rect_sw_on(65,320,0);
131:                 gr_print(70,325,mes20,9,0);
132:                 scope_shatter(1,0);
133:                 scope_shatter(3,0);
134:                 gr_data(data_cnt);
135:                 if(data_cnt==0){
136:                     data_cnt=1;
137:                 }else{
138:                     data_cnt=0;
139:                 }
140:                 rect_sw_off(65,320,0);
141:                 gr_print(70,325,mes20,5,0);
142:                 continue;
143:             }
144:         }
145: 
146:         if(ms_x>150 && ms_x<180){/**/
147:             if(ms_y>320 && ms_y<340){/*windボタン*/
148:                 wind_fnc++;
149:                 if(wind_fnc>3) wind_fnc=0;
150:                 rect_sw_on(150,320,0);
151:                 gr_print(155,325,mes_str5[wind_fnc],9,0);
152:                 rect_sw_off(150,320,0);
153:                 gr_print(155,325,mes_str5[wind_fnc],5,0);
154:                 continue;
155:             }
156:         }
157: 
158:         /*左列ボタン検索*/
159:         if(ms_x>=20 && ms_x<=50){
160: 
161:             if(ms_y>180 && ms_y<200){ /*ｽﾀｰﾄﾎﾞﾀﾝ*/
162:                 rect_sw_on(20,180,0);
163:                 mes_win2(50,180,0);
164:                 gr_print(55,185,mes11,12,0);
165:                 APAGE(3);         /*draw_wave()で強制的にページ3を使用するため iocs とあわせる*/
166:                 draw_wave();
167:                 rect_sw_off(20,180,0);
168:                 mes_win2(50,180,0);
169:                 gr_print(55,185,mes12,12,0);
170:                 continue;
171:             }
172:             if(ms_y>230 && ms_y<250){/*y_scaleボタン*/
173:                 rect_sw_on(20,230,0);
174:                 mes_win2(50,230,0);
175:                 gr_print(55,235,mes_str[j],12,0);
176:                 vol_sift=j;
177:                 rect_sw_off(20,230,0);
178:                 mes_win2(50,230,0);
179:                 gr_print(55,235,mes_str[j],12,0);
180:                 j++;
181:                 if(j>4) j=0;
182:                 continue;
183:             }
184:             if(ms_y>280 && ms_y<300){/*サンプリング周波数*/
185:                 k++;
186:                 if(k>4) k=0;
187:                 rect_sw_on(20,280,0);
188:                 mes_win2(50,280,0);
189:                 gr_print(55,285,mes_str2[k],12,0);
190:                 switch(k){   /*clk=0x80=4MHz,clk=0x00=8MHz */
191:                     case 0:/*3.9KHz*/
192:                         samp=0x03;
193:                         clk=0x80;
194:                         break;
195:                     case 1:/*5.2KHz*/
196:                         samp=0x07;
197:                         clk=0x80;
198:                         break;
199:                     case 2:/*7.8KHz*/
200:                         samp=0x03;
201:                         clk=0x00;
202:                         break;
203:                     case 3:/*10.4KHz*/
204:                         samp=0x07;
205:                         clk=0x00;
206:                         break;
207:                     case 4:/*15.6KHz*/
208:                         samp=0x0b;
209:                         clk=0x00;
210:                         break;
211:                 }
212:                 rect_sw_off(20,280,0);
213:                 mes_win2(50,280,0);
214:                 gr_print(55,285,mes_str2[k],12,0);
215:                 continue;
216:             }
217:             if(ms_y>100 && ms_y<120){/*EXIT switch*/
218:                 rect_sw_on(20,100,0);
219:                 break;
220:             }
221: 
222:             if(ms_y>320 && ms_y<340){/*fft*/
223:                 rect_sw_on(20,320,0);
224:                 gr_print(25,325,mes18,9,0);
225:                 scope_shatter(1,0);
226:                 gr_clr(2);
227:                 gr_clr(3);
228:                 gr_fft(wind_fnc);
229:                 rect_sw_off(20,320,0);
230:                 gr_print(25,325,mes18,5,0);
231:                 continue;
232:             }
233: 
234: 
235: 
236:             /*all x>20 & x<50 end*/
237:         }
238: 
239: /*右列ボタン検索*/
240:         if(ms_x>=110 && ms_x<=140){
241: 
242:             if(ms_y>180 && ms_y<200){ /*calcボタン*/
243:                 rect_sw_on(110,180,0);
244:                 mes_win2(140,180,0);
245:                 gr_print(145,185,mes16,12,0);
246:                 gr_clr(2);
247:                 gr_clr(3);
248:                 calc_wave();
249:                 calc_line(c_pcmbuf);
250:                 rect_sw_off(110,180,0);
251:                 mes_win2(140,180,0);
252:                 gr_print(145,185,mes17,12,0);
253:                 continue;
254:             }
255: 
256:             if(ms_y>230 && ms_y<250){ /*monitorボタン*/
257:                 s++;
```

```
258:                 if(s>3) s=0;
259:                 rect_sw_on(110,230,0);
260:                 mes_win2(140,230,0);
261:                 gr_print(145,235,mes_str4[s],12,0);
262:                 switch(s){
263:                    case 0:
264:                       pan=0x0;
265:                       break;
266:                    case 1:
267:                       pan=0x1;
268:                       break;
269:                    case 2:
270:                       pan=0x2;
271:                       break;
272:                    case 3:
273:                       pan=0x3;
274:                       break;
275:                 }
276:                 rect_sw_off(110,230,0);
277:                 mes_win2(140,230,0);
278:                 gr_print(145,235,mes_str4[s],12,0);
279:                 continue;
280:              }
281:
282:              if(ms_y>280 && ms_y<300){/*glidボタン*/
283:                 l++;
284:                 if(l>2) l=0;
285:                 rect_sw_on(110,280,0);
286:                 mes_win2(140,280,0);
287:                 gr_print(145,285,mes_str3[l],12,0);
288:                 if(l==1){
289:                 scope_shatter(1,0);
290:                 glid2(1);
291:                 }
292:                 if(l==2){
293:                 scope_shatter(1,0);
294:                 }
295:                 if(l==0){
296:                 scope_shatter(1,0);
297:                 glid1(1);
298:                 }
299:                 rect_sw_off(110,280,0);
300:                 mes_win2(140,280,0);
301:                 gr_print(145,285,mes_str3[l],12,0);
302:                 continue;
303:              }
304:              if(ms_y>320 && ms_y<340){/*CLR*/
305:                 rect_sw_on(110,320,0);
306:                 gr_print(115,325,mes19,9,0);
307:                 scope_shatter(2,0);
308:                 scope_shatter(3,0);
309:                 rect_sw_off(110,320,0);
310:                 gr_print(115,325,mes19,5,0);
311:                 continue;
312:              }
313:           }
314:
315:
316:           if(ms_x>=128 && ms_x<=152){                /*position bottan slow 検索*/
317:
318:              if(ms_y>50 && ms_y<74){
319:                 if(h_y>=511) h_y=0;
320:                 tr_sw_on(128,50,0,0);
321:                 h_y++;
322:                 HOME(8,h_x,h_y);
323:                 tr_sw_off(128,50,0,0);
324:                 continue;
325:              }
326:              if(ms_y>98 && ms_y<122){
327:                    if(h_y<=0) h_y=511;
328:                    tr_sw_on(152,122,2,0);
329:                    h_y--;
330:                    HOME(8,h_x,h_y);
331:                    tr_sw_off(152,122,2,0);
332:                    continue;
333:              }
334:           }
335:           if(ms_y>=74 && ms_y<=98){
336:              if(ms_x>104 && ms_x<128){
337:                    if(h_x>=511) h_x=0;
338:                    tr_sw_on(104,98,1,0);
339:                    h_x++;
340:                    HOME(8,h_x,h_y);
341:                    tr_sw_off(104,98,1,0);
342:                    continue;
343:              }
344:              if(ms_x>152 && ms_x<176){
345:                 if(h_x<=0) h_x=511;
346:                    tr_sw_on(176,74,3,0);
347:                    h_x--;
348:                    HOME(8,h_x,h_y);
349:                    tr_sw_off(176,74,3,0);
350:                    continue;
351:              }
352:           }
353:        }
354:           /*右ボタン系*/
355:           if(MS_GETDT()==0x00ff){
356:              ms_temp=MS_CURGT();
357:              ms_y=mask_y & ms_temp;
358:              ms_x=ms_temp>>16;
359:
360:              /*position botton fast 検索*/
361:              if(ms_x>=128 && ms_x<=152){/*上*/
362:                 if(ms_y>50 && ms_y<74){
363:                    if(h_y>=511) h_y=0;
364:                    tr_sw_on(128,50,0,0);
365:                    h_y+=5;
366:                    HOME(8,h_x,h_y);
367:                    tr_sw_off(128,50,0,0);
368:                    continue;
369:                 }
370:                 if(ms_y>98 && ms_y<122){
371:                    if(h_y<=0) h_y=511;
372:                    tr_sw_on(152,122,2,0);
373:                    h_y-=5;
374:                    HOME(8,h_x,h_y);
375:                    tr_sw_off(152,122,2,0);
376:                    continue;
377:                 }
378:              }
379:              if(ms_y>=74 && ms_y<=98){
380:                 if(ms_x>104 && ms_x<128){
381:                    if(h_x>=511) h_x=0;
382:                    tr_sw_on(104,98,1,0);
383:                    h_x+=5;
384:                    HOME(8,h_x,h_y);
385:                    tr_sw_off(104,98,1,0);
386:                    continue;
387:                 }
388:                 if(ms_x>152 && ms_x<176){
389:                    if(h_x<=0) h_x=511;
390:                       tr_sw_on(176,74,3,0);
391:                       h_x-=5;
392:                       HOME(8,h_x,h_y);
393:                       tr_sw_off(176,74,3,0);
394:                       continue;
395:                    }
396:                 }
397:              }
398: /*position bottan 検索終了*/
399:
400:    }/*while end*/
401:
402:    ret_vec();
403:    end_screen();
404: }
405:
406:
407: void    init_screen(){
408:
409:    TGUSEMD(0,2);
410:    TGUSEMD(1,2);
411:    SKEY_MOD(0,0,0);
412:    CRTMOD_ORG=CRTMOD(-1);
413:    CRTMOD(4);
414:    B_CUROFF();
415:    G_CLR_ON();
416:    gpalet_set();
417:    tpalet_set();
418:    MS_INIT();
419:    MS_LIMIT(0,0,511,511);
420:    MS_SEL(0);
421:    MS_CURON();
422: }
423:
424: void    init2_console(){          /*page1にコンソールパネルを描く*/
425:    glid1(1); /*十字カーソル*/
426:    waku1(1);/*外枠*/
427:    panel_main(1);
428:    logo(1);
429: /*各種ボタン& メッセージ表示 初期化page 0*/
430:    rect_sw_off(20,50,0);
431:    rect_sw_off(20,100,0);
432:    mes_win(50,50,0);
433:
434:    gr_print(25,54,mes3,5,0);
435:    gr_print(25,104,mes2,5,0);
436:    gr_print(55,54,mes4,12,0);
437:
438:    rect_sw_off(20,180,0);
439:    rect_sw_off(20,230,0);
440:    rect_sw_off(20,280,0);
441:    rect_sw_off(20,320,0);
442:
443:    rect_sw_off(65,320,0);
444:    rect_sw_off(150,320,0);
445:
446:    rect_sw_off(110,180,0);
447:    rect_sw_off(110,230,0);
448:    rect_sw_off(110,280,0);
449:    rect_sw_off(110,320,0);
450:
451:    mes_win2(50,180,0);
452:    mes_win2(50,230,0);
453:    mes_win2(50,280,0);
454:
455:    mes_win2(140,180,0);
456:    mes_win2(140,230,0);
457:    mes_win2(140,280,0);
458:
459:    gr_print(20,203,mes6,5,0);
460:    gr_print(20,253,mes7,5,0);
461:    gr_print(20,303,mes8,5,0);
462:    gr_print(25,325,mes18,5,0);
463:
464:    gr_print(70,325,mes20,5,0);
465:    gr_print(155,325,mes_str5[0],5,0);
466:
467:    gr_print(110,203,mes15,5,0);
468:    gr_print(110,253,mes9,5,0);
469:    gr_print(110,303,mes10,5,0);
470:    gr_print(115,325,mes19,5,0);
471:
472:    gr_print(116,130,mes14,5,0);
473:    gr_print(55,185,mes12,12,0);
474:    gr_print(55,235,mes_str[3],12,0);
475:    gr_print(55,285,mes_str2[4],12,0);
476:    gr_print(145,185,mes17,12,0);
477:    gr_print(145,235,mes_str4[3],12,0);
478:    gr_print(145,285,mes_str3[0],12,0);
479:    tr_sw_off(128,50,0,0);
480:    tr_sw_off(104,98,1,0);
481:    tr_sw_off(152,122,2,0);
482:    tr_sw_off(176,74,3,0);
483:    waku2(1);
484: }
485:
486: void    end_screen(){
487:    CRTMOD(CRTMOD_ORG);
488:    TGUSEMD(0,0);
489:    TGUSEMD(1,1);
490:    SKEY_MOD(-1,1,1);
491:    C_FNKMOD(0);
492:    C_WINDOW(0,31);
493:    tpalet_ret();
494:    B_CURON();
495:    MS_CUROF();
496:
497: }
498: void    gpalet_set(){
499:    int ssp;
500:    int i;
501:    ssp=SUPER(0);
502:    gpalet     =(unsigned      short *)0xe82000;
503:    for(i=0;i<16;i++){
504:       *gpalet=new_gpalet[i];
505:       gpalet++;
506:    }
507:    SUPER(ssp);
508: }
509:
510: void    scope_shatter(int page,int col){
511:
512:    APAGE(page);
513:    ptr_fill->x1=197;
514:    ptr_fill->y1=7;
515:    ptr_fill->x2=504;
516:    ptr_fill->y2=504;
517:    ptr_fill->color=col;
518:    FILL(ptr_fill);
519: }
520: void    gr_clr(int page){
521:
```

▶とうとう20歳，早く肩書きを見つけないと……。　　和田　智(20)長野県

```
522:     APAGE(page);
523:     ptr_fill->x1=0;
524:     ptr_fill->y1=0;
525:     ptr_fill->x2=511;
526:     ptr_fill->y2=511;
527:     ptr_fill->color=0;
528:     FILL(ptr_fill);
529: }
530:
531: /*x,y の位置に方形のボタンを描く20*30(通常)*/
532: void     rect_sw_off(short x,short y,int nom){
533:
534:     APAGE(nom);
535:
536:     ptr_fill->x1=x;
537:     ptr_fill->y1=y;
538:     ptr_fill->x2=x+30;
539:     ptr_fill->y2=y+20;
540:     ptr_fill->color=7;
541:     FILL(ptr_fill);
542: /*暗影*/
543:     ptr_fill->x1=x+29;
544:     ptr_fill->y1=y;
545:     ptr_fill->x2=x+30;
546:     ptr_fill->y2=y+20;
547:     ptr_fill->color=5;
548:     FILL(ptr_fill);
549:     ptr_fill->x1=x+1;
550:     ptr_fill->y1=y+19;
551:     ptr_fill->x2=x+30;
552:     ptr_fill->y2=y+20;
553:     ptr_fill->color=5;
554:     FILL(ptr_fill);
555: /*明影*/
556:     ptr_fill->x1=x;
557:     ptr_fill->y1=y;
558:     ptr_fill->x2=x+30;
559:     ptr_fill->y2=y+1;
560:     ptr_fill->color=8;
561:     FILL(ptr_fill);
562:     ptr_fill->x1=x;
563:     ptr_fill->y1=y;
564:     ptr_fill->x2=x+1;
565:     ptr_fill->y2=y+20;
566:     ptr_fill->color=8;
567:     FILL(ptr_fill);
568: }
569: /*x,y の位置に方形のボタンを描く20*30(ON)*/
570: void     rect_sw_on(short x,short y,int nom){
571:     int i;
572:     APAGE(nom);
573:
574:     ptr_fill->x1=x;
575:     ptr_fill->y1=y;
576:     ptr_fill->x2=x+30;
577:     ptr_fill->y2=y+20;
578:     ptr_fill->color=6;
579:     FILL(ptr_fill);
580: /*明影*/
581:     ptr_fill->x1=x+29;
582:     ptr_fill->y1=y;
583:     ptr_fill->x2=x+30;
584:     ptr_fill->y2=y+20;
585:     ptr_fill->color=7;
586:     FILL(ptr_fill);
587:     ptr_fill->x1=x+1;
588:     ptr_fill->y1=y+19;
589:     ptr_fill->x2=x+30;
590:     ptr_fill->y2=y+20;
591:     ptr_fill->color=7;
592:     FILL(ptr_fill);
593: /*暗影*/
594:     ptr_fill->x1=x;
595:     ptr_fill->y1=y;
596:     ptr_fill->x2=x+30;
597:     ptr_fill->y2=y+1;
598:     ptr_fill->color=5;
599:     FILL(ptr_fill);
600:     ptr_fill->x1=x;
601:     ptr_fill->y1=y;
602:     ptr_fill->x2=x+1;
603:     ptr_fill->y2=y+20;
604:     ptr_fill->color=5;
605:     FILL(ptr_fill);
606:     for(i=0;i<100000;i++){
607:         int     j;
608:         j+=i;
609:     }
610: }
611: /*x,y の位置に長方形のメッセージ窓を描く20*50*/
612: void     mes_win(short x,short y,int nom){
613:
614:     APAGE(nom);
615:
616:     ptr_fill->x1=x;
617:     ptr_fill->y1=y;
618:     ptr_fill->x2=x+50;
619:     ptr_fill->y2=y+20;
620:     ptr_fill->color=7;
621:     FILL(ptr_fill);
622: /*明影*/
623:     ptr_fill->x1=x+50;
624:     ptr_fill->y1=y;
625:     ptr_fill->x2=x+50;
626:     ptr_fill->y2=y+20;
627:     ptr_fill->color=8;
628:     FILL(ptr_fill);
629:     ptr_fill->x1=x+1;
630:     ptr_fill->y1=y+20;
631:     ptr_fill->x2=x+50;
632:     ptr_fill->y2=y+20;
633:     ptr_fill->color=8;
634:     FILL(ptr_fill);
635: /*暗影*/
636:     ptr_fill->x1=x;
637:     ptr_fill->y1=y;
638:     ptr_fill->x2=x+50;
639:     ptr_fill->y2=y;
640:     ptr_fill->color=5;
641:     FILL(ptr_fill);
642:     ptr_fill->x1=x;
643:     ptr_fill->y1=y;
644:     ptr_fill->x2=x;
645:     ptr_fill->y2=y+20;
646:     ptr_fill->color=5;
647:     FILL(ptr_fill);
648: }
649: void     mes_win2(short x,short y,int nom){
650:
651:     APAGE(nom);
652:     ptr_fill->x1=x;
653:     ptr_fill->y1=y;
654:     ptr_fill->x2=x+30;
655:     ptr_fill->y2=y+20;
656:     ptr_fill->color=7;
657:     FILL(ptr_fill);
658: /*明影*/
659:     ptr_fill->x1=x+30;
660:     ptr_fill->y1=y;
661:     ptr_fill->x2=x+30;
662:     ptr_fill->y2=y+20;
663:     ptr_fill->color=8;
664:     FILL(ptr_fill);
665:     ptr_fill->x1=x+1;
666:     ptr_fill->y1=y+20;
667:     ptr_fill->x2=x+30;
668:     ptr_fill->y2=y+20;
669:     ptr_fill->color=8;
670:     FILL(ptr_fill);
671: /*暗影*/
672:     ptr_fill->x1=x;
673:     ptr_fill->y1=y;
674:     ptr_fill->x2=x+30;
675:     ptr_fill->y2=y;
676:     ptr_fill->color=5;
677:     FILL(ptr_fill);
678:     ptr_fill->x1=x;
679:     ptr_fill->y1=y;
680:     ptr_fill->x2=x;
681:     ptr_fill->y2=y+20;
682:     ptr_fill->color=5;
683:     FILL(ptr_fill);
684: }
685:
686: /*グラフィックに文字列を表示する*/
687: void     gr_print(short x,short y,unsigned char *mes,short col,int page){
688:
689:     APAGE(page);
690:     ptr_symbol->x1=x;
691:     ptr_symbol->y1=y;
692:     ptr_symbol->string_address=mes;
693:     ptr_symbol->mag_x=1;
694:     ptr_symbol->mag_y=1;
695:     ptr_symbol->color=col;
696:     ptr_symbol->font_type=0;
697:     ptr_symbol->angle=0;
698:     SYMBOL(ptr_symbol);
699: }
700: void     glid1(int page){
701:     int i;
702:             APAGE(page);
703:             ptr_line->x1=350;
704:             ptr_line->y1=8;
705:             ptr_line->x2=350;
706:             ptr_line->y2=503;
707:             ptr_line->color=2;
708:             ptr_line->linestyle=0xffff;
709:             LINE(ptr_line);
710: /*十字カーソル横*/
711:             ptr_line->x1=196;
712:             ptr_line->y1=260;
713:             ptr_line->x2=506;
714:             ptr_line->y2=260;
715:             ptr_line->color=2;
716:             ptr_line->linestyle=0xffff;
717:             LINE(ptr_line);
718: /*目盛り*/
719:             for(i=0;i<=504;i+=14){
720:                 ptr_line->x1=(196+i);
721:                 ptr_line->y1=255;
722:                 ptr_line->x2=(196+i);
723:                 ptr_line->y2=265;
724:                 ptr_line->color=2;
725:                 ptr_line->linestyle=0xffff;
726:                 LINE(ptr_line);
727:                 ptr_line->x1=345;
728:                 ptr_line->y1=(8+i);
729:                 ptr_line->x2=355;
730:                 ptr_line->y2=(8+i);
731:                 ptr_line->color=2;
732:                 ptr_line->linestyle=0xffff;
733:                 LINE(ptr_line);
734:             }
735: }
736: void     glid2(int page){
737:     int i;
738:             APAGE(page);
739: /*glid*/
740:             for(i=0;i<=504;i+=14){
741:                 ptr_line->x1=(196+i);
742:                 ptr_line->y1=8;
743:                 ptr_line->x2=(196+i);
744:                 ptr_line->y2=503;
745:                 ptr_line->color=3;
746:                 ptr_line->linestyle=0x4444;
747:                 LINE(ptr_line);
748:                 ptr_line->x1=196;
749:                 ptr_line->y1=(8+i);
750:                 ptr_line->x2=506;
751:                 ptr_line->y2=(8+i);
752:                 ptr_line->color=3;
753:                 ptr_line->linestyle=0x4444;
754:                 LINE(ptr_line);
755:             }
756: /*十字カーソル縦*/
757:             ptr_line->x1=350;
758:             ptr_line->y1=8;
759:             ptr_line->x2=350;
760:             ptr_line->y2=503;
761:             ptr_line->color=3;
762:             ptr_line->linestyle=0xffff;
763:             LINE(ptr_line);
764: /*十字カーソル横*/
765:             ptr_line->x1=196;
766:             ptr_line->y1=260;
767:             ptr_line->x2=503;
768:             ptr_line->y2=260;
769:             ptr_line->color=3;
770:             ptr_line->linestyle=0xffff;
771:             LINE(ptr_line);
772: }
773: void     waku1(int page){
774:
775:     APAGE(page);
776:     ptr_fill->x1=190;
777:     ptr_fill->y1=0;
778:     ptr_fill->x2=511;
779:     ptr_fill->y2=2;
780:     ptr_fill->color=4;
781:     FILL(ptr_fill);
782:     ptr_fill->x1=190;
783:     ptr_fill->y1=0;
784:     ptr_fill->x2=192;
785:     ptr_fill->y2=511;
```

▶国民機を買う予定だったお金が免許になりそう。「正しいお金の使い方だよな」と自分を納得させつつ，「これだったらXellent30買って，MO買って……」なんて思ったりもする。でも本当は可愛い女のことTDLに遊びに行きたいよー！　誰か私にもう1個(?)夏休みをくれ。
西川　和範(20)東京都

```
786:    ptr_fill->color=4;
787:    FILL(ptr_fill);
788:    ptr_fill->x1=509;
789:    ptr_fill->y1=0;
790:    ptr_fill->x2=511;
791:    ptr_fill->y2=511;
792:    ptr_fill->color=4;
793:    FILL(ptr_fill);
794:    ptr_fill->x1=190;
795:    ptr_fill->y1=509;
796:    ptr_fill->x2=511;
797:    ptr_fill->y2=511;
798:    ptr_fill->color=4;
799:    FILL(ptr_fill);
800:
801:    ptr_fill->x1=195;
802:    ptr_fill->y1=3;
803:    ptr_fill->x2=509;
804:    ptr_fill->y2=7;
805:    ptr_fill->color=3;
806:    FILL(ptr_fill);
807:    ptr_fill->x1=193;
808:    ptr_fill->y1=3;
809:    ptr_fill->x2=197;
810:    ptr_fill->y2=509;
811:    ptr_fill->color=3;
812:    FILL(ptr_fill);
813:
814:    ptr_fill->x1=504;
815:    ptr_fill->y1=6;
816:    ptr_fill->x2=509;
817:    ptr_fill->y2=507;
818:    ptr_fill->color=2;
819:    FILL(ptr_fill);
820:    ptr_fill->x1=195;
821:    ptr_fill->y1=504;
822:    ptr_fill->x2=509;
823:    ptr_fill->y2=509;
824:    ptr_fill->color=2;
825:    FILL(ptr_fill);
826: }
827: void    panel_main(int page){
828:    APAGE(page);
829: /*パネル面*/
830:    ptr_fill->x1=0;
831:    ptr_fill->y1=0;
832:    ptr_fill->x2=189;
833:    ptr_fill->y2=511;
834:    ptr_fill->color=6;
835:    FILL(ptr_fill);
836: /*明影上*/
837:    ptr_fill->x1=0;
838:    ptr_fill->y1=21;
839:    ptr_fill->x2=189;
840:    ptr_fill->y2=26;
841:    ptr_fill->color=7;
842:    FILL(ptr_fill);
843:    ptr_fill->x1=0;
844:    ptr_fill->y1=21;
845:    ptr_fill->x2=5;
846:    ptr_fill->y2=160;
847:    ptr_fill->color=7;
848:    FILL(ptr_fill);
849: /*暗影上*/
850:    ptr_fill->x1=184;
851:    ptr_fill->y1=25;
852:    ptr_fill->x2=189;
853:    ptr_fill->y2=160;
854:    ptr_fill->color=5;
855:    FILL(ptr_fill);
856:    ptr_fill->x1=0;
857:    ptr_fill->y1=155;
858:    ptr_fill->x2=189;
859:    ptr_fill->y2=160;
860:    ptr_fill->color=5;
861:    FILL(ptr_fill);
862: /*明影下*/
863:    ptr_fill->x1=0;
864:    ptr_fill->y1=161;
865:    ptr_fill->x2=189;
866:    ptr_fill->y2=166;
867:    ptr_fill->color=7;
868:    FILL(ptr_fill);
869:    ptr_fill->x1=0;
870:    ptr_fill->y1=161;
871:    ptr_fill->x2=5;
872:    ptr_fill->y2=511;
873:    ptr_fill->color=7;
874:    FILL(ptr_fill);
875: /*暗影下*/
876:    ptr_fill->x1=184;
877:    ptr_fill->y1=165;
878:    ptr_fill->x2=189;
879:    ptr_fill->y2=511;
880:    ptr_fill->color=5;
881:    FILL(ptr_fill);
882:    ptr_fill->x1=0;
883:    ptr_fill->y1=506;
884:    ptr_fill->x2=189;
885:    ptr_fill->y2=511;
886:    ptr_fill->color=5;
887:    FILL(ptr_fill);
888:
889: }
890: void    logo(int page){
891:    APAGE(page);
892: /*ロゴパネル*/
893:    ptr_fill->x1=0;
894:    ptr_fill->y1=0;
895:    ptr_fill->x2=189;
896:    ptr_fill->y2=20;
897:    ptr_fill->color=10;
898:    FILL(ptr_fill);
899: /*明影*/
900:    ptr_fill->x1=0;
901:    ptr_fill->y1=0;
902:    ptr_fill->x2=189;
903:    ptr_fill->y2=1;
904:    ptr_fill->color=11;
905:    FILL(ptr_fill);
906:    ptr_fill->x1=0;
907:    ptr_fill->y1=0;
908:    ptr_fill->x2=1;
909:    ptr_fill->y2=20;
910:    ptr_fill->color=11;
911:    FILL(ptr_fill);
912: /*暗影*/
913:    ptr_fill->x1=188;
914:    ptr_fill->y1=0;
915:    ptr_fill->x2=189;
916:    ptr_fill->y2=20;
917:    ptr_fill->color=9;
918:    FILL(ptr_fill);
919:    ptr_fill->x1=0;
920:    ptr_fill->y1=19;
921:    ptr_fill->x2=189;
922:    ptr_fill->y2=20;
923:    ptr_fill->color=9;
924:    FILL(ptr_fill);
925:
926:    gr_print(4,4,mes1,12,0);
927: }
928:
929: void    tr_sw_off(short x,short y,short ang,int page){
930:
931:    APAGE(page);
932:    ptr_symbol->x1=x;
933:    ptr_symbol->y1=y;
934:    ptr_symbol->string_address=mes13;
935:    ptr_symbol->mag_x=1;
936:    ptr_symbol->mag_y=1;
937:    ptr_symbol->color=8;
938:    ptr_symbol->font_type=2;
939:    ptr_symbol->angle=ang;
940:    SYMBOL(ptr_symbol);
941:    ptr_symbol->x1=x+4;
942:    ptr_symbol->y1=y;
943:    ptr_symbol->string_address=mes13;
944:    ptr_symbol->mag_x=1;
945:    ptr_symbol->mag_y=1;
946:    ptr_symbol->color=5;
947:    ptr_symbol->font_type=2;
948:    ptr_symbol->angle=ang;
949:    SYMBOL(ptr_symbol);
950:    ptr_symbol->x1=x+2;
951:    ptr_symbol->y1=y;
952:    ptr_symbol->string_address=mes13;
953:    ptr_symbol->mag_x=1;
954:    ptr_symbol->mag_y=1;
955:    ptr_symbol->color=7;
956:    ptr_symbol->font_type=2;
957:    ptr_symbol->angle=ang;
958:    SYMBOL(ptr_symbol);
959: }
960: void    tr_sw_on(short x,short y,short ang,int page){
961:    APAGE(page);
962:    ptr_symbol->x1=x;
963:    ptr_symbol->y1=y;
964:    ptr_symbol->string_address=mes13;
965:    ptr_symbol->mag_x=1;
966:    ptr_symbol->mag_y=1;
967:    ptr_symbol->color=5;
968:    ptr_symbol->font_type=2;
969:    ptr_symbol->angle=ang;
970:    SYMBOL(ptr_symbol);
971:    ptr_symbol->x1=x+4;
972:    ptr_symbol->y1=y;
973:    ptr_symbol->string_address=mes13;
974:    ptr_symbol->mag_x=1;
975:    ptr_symbol->mag_y=1;
976:    ptr_symbol->color=8;
977:    ptr_symbol->font_type=2;
978:    ptr_symbol->angle=ang;
979:    SYMBOL(ptr_symbol);
980:    ptr_symbol->x1=x+2;
981:    ptr_symbol->y1=y;
982:    ptr_symbol->string_address=mes13;
983:    ptr_symbol->mag_x=1;
984:    ptr_symbol->mag_y=1;
985:    ptr_symbol->color=6;
986:    ptr_symbol->font_type=2;
987:    ptr_symbol->angle=ang;
988:    SYMBOL(ptr_symbol);
989: }
990:
991: void    ret_vec(){
992:    INTVCS(break_key,(char *)org4_vector);
993:    INTVCS(prn_break,(char *)org3_vector);
994: /*      INTVCS(intrapt_sw,(char *)org2_vector);*/
995:    INTVCS(niv_no,(char *)org1_vector);
996: }
997: /* void er_break(){ インターラプトスイッチが押された時の終了処理
998:    int ssp;
999:    dma_stop();
1000:    end_screen();
1001:    ret_vec();
1002:    EXIT();
1003: }
1004: */
1005: void    hd_copy(){
1006:    __builtin_saveregs();
1007:    /*単なる rte */
1008: }
1009: void    waku2(int page){
1010:
1011:    APAGE(page);
1012:    ptr_fill->x1=10;
1013:    ptr_fill->y1=350;
1014:    ptr_fill->x2=180;
1015:    ptr_fill->y2=501;
1016:    ptr_fill->color=7;
1017:    FILL(ptr_fill);
1018:    ptr_fill->x1=10;
1019:    ptr_fill->y1=350;
1020:    ptr_fill->x2=180;
1021:    ptr_fill->y2=352;
1022:    ptr_fill->color=1;
1023:    FILL(ptr_fill);
1024:    ptr_fill->x1=178;
1025:    ptr_fill->y1=350;
1026:    ptr_fill->x2=180;
1027:    ptr_fill->y2=501;
1028:    ptr_fill->color=3;
1029:    FILL(ptr_fill);
1030:    ptr_fill->x1=10;
1031:    ptr_fill->y1=499;
1032:    ptr_fill->x2=180;
1033:    ptr_fill->y2=501;
1034:    ptr_fill->color=3;
1035:    FILL(ptr_fill);
1036:    ptr_fill->x1=10;
1037:    ptr_fill->y1=350;
1038:    ptr_fill->x2=12;
1039:    ptr_fill->y2=501;
1040:    ptr_fill->color=1;
1041:    FILL(ptr_fill);
1042:
1043: }
1044: void    tpalet_set(){ /*text plane 1 =青 text plane 2=オレンジ*/
1045:    int ssp;
1046:    int i;
1047:    unsigned short *temp;
1048:    ssp=SUPER(0);
1049:    tpalet    =(unsigned    short *)0xe82200;
1050:    temp=tpalet;
1051:    for(i=0;i<16;i++){
```

▶車に乗ってOh!Xを買いに本屋へ行く途中で信号待ちになった。青になっても前の車が発車しない。まぁ急いでいないからと，クラクションを鳴らさないでいたらついに赤になった。いったいなんてやつだ！　次に青になっても動かないので追い抜いたら車の前がペシャンコ！　事故車を交差点に置いとくな！　　市川　博基(20)愛知県

```
1052:         text_palet[i]=*(temp++);
1053:     }
1054:     *(tpalet+1)=0x8e81;
1055:     *(tpalet+2)=0x3059;
1056:     SUPER(ssp);
1057: }
1058:
1059: void    tpalet_ret(){
1060:     int ssp;
```

```
1061:     int i;
1062:     ssp=SUPER(0);
1063:     tpalet      =(unsigned      short *)0xe82200;
1064:     for(i=0;i<16;i++){
1065:         *(tpalet++)=text_palet[i];
1066:     }
1067:     SUPER(ssp);
1068: }
```

リスト2 DRAW WAVE.C

```
  1: #include        "wave.h"
  2:
  3:
  4: extern  signed  short   pcmbuf[PCMSIZE];
  5: extern  int     CRTMOD_ORG;
  6: extern  int     wind_fnc;
  7: extern  unsigned char   samp;
  8: extern  unsigned char   clk;
  9: extern  unsigned char   pan;
 10: extern struct FILLPTR   *ptr_fill;
 11: extern struct LINEPTR   *ptr_line;
 12:
 13: volatile struct    DMAREG *dma          =(struct  DMAREG *)0xe840c0;
 14: volatile unsigned char    *ppi_c        =(unsigned      char *)0xe9a005;
 15: volatile unsigned char    *opm_regno    =(unsigned      char *)0xe90001;
 16: volatile unsigned char    *opm_data     =(unsigned      char *)0xe90003;
 17: volatile unsigned char    *adpcm_command=(unsigned      char *)0xe92001;
 18: volatile unsigned char    *adpcm_status =(unsigned      char *)0xe92001;
 19: volatile unsigned char    *adpcm_data   =(unsigned      char *)0xe92003;
 20:
 21: volatile unsigned         char    inp_frag;
 22: volatile signed           short   *last_pcm_add;
 23: volatile signed           short   last_val;
 24: volatile signed           short   val_x;`
 25: volatile unsigned         short   len;
 26: volatile unsigned         short   len_pcm_buf;
 27: volatile unsigned         short   pcm_len=PCMSIZE;
 28:
 29: unsigned char   ad_pcmbuf1[BUFSIZE];
 30: unsigned char   ad_pcmbuf2[BUFSIZE];
 31:
 32: signed   short  c_pcmbuf[15600]; /*1秒分*/
 33: unsigned char   c_adpcmbuf[7800];
 34: unsigned short          c_len=7800;
 35:
 36: CMPLX   c[BUFF_COMP+1];/*fftの変換結果*/
 37: CMPLX   y[BUFF_COMP+1];/*元のデーター*/
 38: CMPLX   *p;
 39:
 40: void    draw_wave(){    /*リアルタイム用*/
 41:
 42:     int ssp;
 43:     /****割り込み処理用変数、最初の初期化****/
 44:     inp_frag=0;
 45:     last_val=0;
 46:     val_x=0;
 47:     last_pcm_add=pcmbuf;
 48:     len=BUFSIZE;
 49:     len_pcm_buf=PCMSIZE;
 50:     /*****************************************/
 51:     /*ここから先はスーパーバイザーモード*/
 52:     /*  warikomi=dma ad_pcmdata転送(継続動作モード)+リアルタイムad_pcm->pcm変換
 53:     Cの関数として使用
 54:     */
 55:     ssp=SUPER(0);
 56:
 57:     *adpcm_data=0x00;
 58:
 59:     dma_stop();
 60:     adpcm_sample(samp,clk,pan);
 61:     clear_flag();/*dma csr,cer初期化*/
 62:     dma_setup();
 63:     dma_start();
 64:     draw_mod();
 65:     dma_stop();
 66:     SUPER(ssp);
 67: }
 68:
 69:
 70: /*      last_pcm_add=割り込みにより書き換えが終了している現在のpcmbufアドレスpointer
 71:     len_pcm_buf=書き換えが終了しているpcmbufから終わりまでの残りデーター個数=short(*2=byte)
 72:     pcm_len=PCMSIZE=pcmbufの大きさ(shortsize)
 73: */
 74: void    calc_wave(){/*波形解析用*/
 75:     char        err_cord;
 76:     int count1=0;
 77:     int count2=0;
 78:     int i;
 79:     int max=0;
 80:     int min=0;
 81:     int cnt_flg=0;
 82:     double freq=0.0;
 83:     double rms=0.0;
 84:     double heikin=0.0;
 85:     double temp_rms=0.0;
 86:     double temp_heikin=0.0;
 87:
 88:     B_LOCATE(2,21);
 89:     B_CLR_ED();
 90:     B_CUROFF();
 91:
 92: /*取敢えず1秒分計測する*/
 93:     err_cord=dma_adpcm(c_adpcmbuf,c_len);
 94:
 95:     if(err_cord>0){
 96:         printf("dma_err!!¥n");
 97:         return;
 98:     }
 99:     ad_to_pcm(c_pcmbuf,c_adpcmbuf,(int)c_len);
100:
101:         max=min=c_pcmbuf[0];
102:
103:     for(i=0;i<(c_len*2);i++){
104:
105:         if(min>c_pcmbuf[i]) min=c_pcmbuf[i];
106:         if(max<c_pcmbuf[i]) max=c_pcmbuf[i];
107:
108:         if(c_pcmbuf[i]<0 && c_pcmbuf[i+1]>=0){
109:             if(cnt_flg==0){
110:             count1=i;
111:             cnt_flg+=1;
112:             }
113:         }
114:         if(cnt_flg==1){
115:             heikin=rms=c_pcmbuf[i+1];
116:             rms*=rms;
117:             if(c_pcmbuf[i+1]>=0){
118:             temp_heikin+=heikin;
119:             }else{
120:             temp_heikin+=(-1*heikin);
121:             }
122:             temp_rms+=rms;
123:             if(c_pcmbuf[i]>0 && c_pcmbuf[i+1]<=0){
124:             count2=i+1;
125:             count1=(i-count1)+1;
126:             cnt_flg+=1;
127:             continue;
128:             }
129:         }
130:         if(cnt_flg==2){
131:             heikin=rms=c_pcmbuf[i+1];
132:             rms*=rms;
133:             if(c_pcmbuf[i+1]>=0){
134:             temp_heikin+=heikin;
135:             }else{
136:             temp_heikin+=(-1*heikin);
137:             }
138:             temp_rms+=rms;
139:             if(c_pcmbuf[i]<0 && c_pcmbuf[i+1]>=0){
140:             count2=(i-count2)+1;
141:             break;
142:             }
143:         }
144:
145:     }
146:         printf("s");
147:         printf("31m");
148:         B_LOCATE(2,23);
149:         printf("最大値=%+d¥n",max);
150:         B_LOCATE(2,24);
151:         printf("最小値=%+d¥n",min);
152:
153:     if(cnt_flg!=2){
154:         B_LOCATE(2,25);
155:         printf("周波数が1Hz以下¥n");
156:         B_LOCATE(2,26);
157:         printf("又は計測不可能の¥n");
158:         B_LOCATE(2,27);
159:         printf("信号レベル、¥n");
160:         B_LOCATE(2,28);
161:         printf("周波数です¥n");
162:         printf("u");
163:         B_CUROFF();
164:         return;
165:     }else{
166:         freq=(double)(15625.0/(count1+count2-1));
167:         if(clk==0x80){
168:             if(samp==0x03) freq/=4.0;
169:             if(samp==0x07) freq/=3.0;
170:         }
171:         if(clk==0x00){
172:             if(samp==0x03) freq/=2.0;
173:             if(samp==0x07) freq/=1.5;
174:         }
175:
176:         B_LOCATE(2,25);
177:         printf("周波数=%-.2f[Hz]¥n",freq);
178:         rms=100*(sqrt(temp_rms/(count1+count2)))/max;
179:         B_LOCATE(2,26);
180:         printf("実効値=%-.2f%¥n",rms);
181:         heikin=100*(temp_heikin/(count1+count2))/max;
182:         B_LOCATE(2,27);
183:         printf("平均値=%-.2f%¥n",heikin);
184:       . temp_heikin=rms/heikin;
185:         B_LOCATE(2,28);
186:         printf("波形率=%-.2f¥n",temp_heikin);
187:         temp_rms=1/(rms/100);
188:         B_LOCATE(2,29);
189:         printf("波高率=%-.2f¥n",temp_rms);
190:
191:     }
192:     printf("u");
193:     B_CUROFF();
194: }
195:
196: char    dma_adpcm(unsigned char *ad_add,short ad_len){/*ストレージ用*/
197:
198:     int ssp;
199:     char err_code;
200:     /*ここから先はスーパーバイザーモード*/
201:     /*dma ad_pcmdata転送(通常動作モード) */
202:
203:     ssp=SUPER(0);
204:     dma_stop();
205:     adpcm_sample(samp,clk,pan);
206:     clear_flag();                                   /*dma csr,cer初期化*/
207:     dma_set_normal(ad_add,ad_len);                  /*adpcm_bufに転送*/
208:     dma_start2();
209:     while(!(dma->csr & 0x90));
210:     err_code=dma->cer;
211:     if(err_code>0) return(err_code);
212:     dma_stop();
213:     SUPER(ssp);
214:     return(0);
215: }
216:
217: void    adpcm_sample(unsigned char samp,unsigned char clk,unsigned char pan){
218:
219:     *opm_regno=0x1b;
220:     *opm_data=clk;
221:     *ppi_c=samp;
222:     *ppi_c &=0xc;
223:     *ppi_c |=pan;
224: }
225:
226:
227: void    dma_setup(){ /*継続動作  初期化*/
228:
229:     dma->dcr=0x80;
230:     dma->ocr=0xb2; /*デバイス−>メモリ*/
231:     dma->scr=0x04;
232:     dma->ccr=0x00; /*08 割り込み許可*/
233:     dma->cpr=0x00;
234:     dma->mfc=0x05;/*ファンクションコード=スーパーバイザーデータ*/
235:     dma->dfc=0x05;
236:     dma->bfc=0x05;
237:     dma->mar=ad_pcmbuf1;
238:     dma->mtc=len;
```

▶次世代マシンは必ず出してもらわなければ困ります。もうシャープのパソコン以外は受け付けない身体になってしまいました(赤面)。責任とってください。

伊藤　義博(25)東京都

```
239:     dma->btc=len;
240:     dma->dar=(unsigned char *)adpcm_data;
241:     dma->bar=ad_pcmbuf2;
242:
243: }
244: void    dma_set_normal(unsigned char *ad_add,short buf_len){ /*通常動作  初期化*/
245:
246:     dma->dcr=0x80;
247:     dma->ocr=0xb2; /*デバイスー>メモリ*/
248:     dma->scr=0x04;
249:     dma->ccr=0x00;
250:     dma->cpr=0x00;
251:     dma->mfc=0x05;/*ファンクションコード=スーパーバイザーデータ*/
252:     dma->dfc=0x05;
253:     dma->bfc=0x05;
254:     dma->mar=ad_add;
255:     dma->mtc=buf_len;
256:     dma->dar=(unsigned char *)adpcm_data;
257:
258: }
259:
260: void    dma_start(){
261:
262:     *adpcm_command=0x4; /*録音*/
263:     dma->ccr =0x88;                /*割り込みon & start*/
264:     dma->ccr |=0x40;  /*cont許可*/
265: }
266:
267: void    dma_start2(){
268:
269:     *adpcm_command=0x4; /*録音*/
270:     dma->ccr =0x80;                /*割り込み禁止 & start*/
271: }
272:
273: void    dma_stop(){
274:     *adpcm_command=0x01;
275:     dma->ccr =0x10;
276: }
277:
278:
279: void    clear_flag(){
280:     dma->csr=0xff;
281:     dma->ccr=0x0;
282: }
283:
284: void    gr_fft(int wind){
285: /*FFT!! */
286:     int i,l,cnt,j,k,x0,x1,y0,y1;
287:     int bar_sp=0;
288:     int skip_x=1;
289:     signed short dat1;
290:     signed short dat2;
291:
292:     for(i=0;i<(c_len*2);i++){
293:        if(c_pcmbuf[i]<0 && c_pcmbuf[i+1]>=0) break;
294:     }
295:     if(i>((c_len*2)-BUFF_COMP)) i=0;
296:     j=0;
297:     k=i;
298:     ptr_line->color=14;
299:     ptr_line->linestyle=0xffff;
300:
301:     switch(wind){
302:        case 0:
303:              rect(i,j);
304:              break;
305:        case 1:
306:              hann(i,j);
307:              break;
308:        case 2:
309:              blac(i,j);
310:              break;
311:        case 3:
312:              rose(i,j);
313:              break;
314:     }
315:
316:     APAGE(2);
317:     j=k+2;
318:     dat1=c_pcmbuf[k]/8;
319:     dat2=c_pcmbuf[k+1]/8;
320:     cnt=1;
321:     x0=197;
322:     for(l=k;l<(k+320);l++){
323:        if(skip_x==2){
324:           j++;
325:           skip_x=1;
326:        }
327:        y0=255-dat1;
328:        if(y0<0) y0=0;
329:        if(y0>512) y0=512;
330:        y1=255-dat2;
331:        x1=x0+1;
332:        ptr_line->x1=x0;
333:        ptr_line->y1=y0;
334:        ptr_line->x2=x1;
335:        ptr_line->y2=y1;
336:        LINE(ptr_line);
337:
338:        if(cnt==256) ptr_line->color=10;
339:        cnt++;
340:        dat1=dat2;
341:        dat2=c_pcmbuf[j]/8;
342:        x0++;
343:        j++;
344:        skip_x++;
345:     }
346:     gr_print(200,20,"FFT 計算中***       ",13,2);
347:     fft(c,y,BUFF_COMP,-1);
348:     for(i=1;i<=50;i++){
349:        p=&c[i];
350:        y[i].x=cabs(p);
351:     }
352:     ptr_fill->x1=200;
353:     ptr_fill->y1=20;
354:     ptr_fill->x2=500;
355:     ptr_fill->y2=60;
356:     ptr_fill->color=0;
357:     FILL(ptr_fill);
358:     gr_print(200,20," [ POWER SPECTUM RESULTS ] ",13,2);
359:     gr_print(200,40,"   DATA POINT 512 ,BASE FREQUENCY=30.52 Hz ",13,2);
360:     for(i=1;i<=50;i++){
361:        x0=200+bar_sp;
362:        x1=x0+3;
363:        y0=490-(int)(y[i].x);
364:        if(y0<0) y0=0;
365:        y1=490;
366:     ptr_fill->x1=x0;
367:     ptr_fill->y1=y0;
368:     ptr_fill->x2=x1;
369:     ptr_fill->y2=y1;
370:     ptr_fill->color=13;
371:     FILL(ptr_fill);
372:     bar_sp+=6;
373:     }
374: }
375:
376: void    calc_line(signed short *pcm_line){/*pcm_lineは620ポイント分以上*/
377:     signed short dat1;
378:     signed short dat2;
379:     int i,l,j,x0,x1,y0,y1;
380:     int skip_x=1;
381:     int cnt;
382:     APAGE(2);
383:     x0=197;
384:     ptr_line->color=14;
385:     ptr_line->linestyle=0xffff;
386:     gr_print(400,490,"【計測対象波形】",13;2);
387:     for(i=0;i<(c_len*2);i++){
388:        if(pcm_line[i]<0 && pcm_line[i+1]>=0) break;
389:     }
390:     if(i>((c_len*2)-650)) i=0;/*620ポイント分描く(一つ置にデーターを取る)*/
391:     j=i+2;
392:     dat1=pcm_line[i]/8;
393:     dat2=pcm_line[i+1]/8;
394:     cnt=1;
395:     for(l=i;l<(i+320);l++){
396:        if(skip_x==2){
397:           j++;
398:           skip_x=1;
399:        }
400:
401:        y0=255-dat1;
402:        if(y0<0) y0=0;
403:        if(y0>512) y0=512;
404:        y1=255-dat2;
405:        x1=x0+1;
406:        ptr_line->x1=x0;
407:        ptr_line->y1=y0;
408:        ptr_line->x2=x1;
409:        ptr_line->y2=y1;
410:        LINE(ptr_line);
411:
412:        cnt++;
413:        dat1=dat2;
414:        dat2=pcm_line[j]/8;
415:        x0++;
416:        j++;
417:        skip_x++;
418:     }
419: }
420:
421: void    gr_data(int cnt){
422:
423:     int i;
424:     int x_offset=220;
425:     int y_offset=80;
426:     int y_space=15;
427:     int x_space=70;
428:     char real[16];
429:     char image[16];
430:     char val[16];
431:     APAGE(2);
432:     ptr_fill->x1=200;
433:     ptr_fill->y1=20;
434:     ptr_fill->x2=500;
435:     ptr_fill->y2=60;
436:     ptr_fill->color=0;
437:     FILL(ptr_fill);
438:        gr_print(200,20," [ POWER SPECTUM RESULTS ] ",13,1);
439:        gr_print(200,40,"   DATA POINT 512 ,BASE FREQUENCY=30.52 Hz ",13,1);
440:     if(cnt==0){
441:           gr_print(x_offset,60,"real",13,1);
442:           gr_print(x_offset+x_space,60,"image",13,1);
443:        for(i=1;i<=28;i++){
444:           (void)gcvt(c[i].x,4,real);
445:           (void)gcvt(c[i].y,4,image);
446:           (void)itoa(i-1,val,10);
447:           gr_print(x_offset-20,y_offset,val,13,1);
448:           gr_print(x_offset,y_offset,real,12,1);
449:           gr_print(x_offset+x_space,y_offset,image,12,1);
450:           y_offset+=15;
451:        }
452:           x_offset=370;
453:           y_offset=80;
454:           gr_print(x_offset,60,"real",13,1);
455:           gr_print(x_offset+x_space,60,"image",13,1);
456:        for(i=29;i<=50;i++){
457:           (void)gcvt(c[i].x,4,real);
458:           (void)gcvt(c[i].y,4,image);
459:           (void)itoa(i-1,val,10);
460:           gr_print(x_offset-20,y_offset,val,13,1);
461:           gr_print(x_offset,y_offset,real,12,1);
462:           gr_print(x_offset+x_space,y_offset,image,12,1);
463:           y_offset+=15;
464:        }
465:     }else{
466:           gr_print(x_offset,60,"power",13,1);
467:        for(i=1;i<=28;i++){
468:           (void)gcvt(y[i].x,4,real);
469:           (void)itoa(i-1,val,10);
470:           gr_print(x_offset-20,y_offset,val,13,1);
471:           gr_print(x_offset,y_offset,real,12,1);
472:           y_offset+=15;
473:        }
474:           x_offset=370;
475:           y_offset=80;
476:           gr_print(x_offset,60,"power",13,1);
477:        for(i=29;i<=50;i++){
478:           (void)gcvt(y[i].x,4,real);
479:           (void)itoa(i-1,val,10);
480:           gr_print(x_offset-20,y_offset,val,13,1);
481:           gr_print(x_offset,y_offset,real,12,1);
482:           y_offset+=15;
483:        }
484:
485:     }
486: }
487:
488: void    rect(int i,int j){
489:     int k=i;
490:     for(;i<(k+BUFF_COMP);i++){
491:        y[j+1].x=(double)c_pcmbuf[i];
492:        y[j+1].y=(double)0.0;
493:        j++;
494:     }
495: }
496:
497: void    hann(int i,int j){
498:     int k=i;
499:     for(;i<(k+BUFF_COMP);i++){
500:        y[j+1].x=(double)c_pcmbuf[i]*(0.5-0.5*cos(2.0*PI*i/512));
501:        y[j+1].y=(double)0.0;
502:        j++;
503:     }
504: }
```

▶暑くてへばってます。68君もクロックアップしてへばっているのかな？　クーラーをつけてあげよー。　藍原　和久(24)東京都

```
505: void    blac(int i,int j){
506:     int k=i;
507:     for(;i<(k+BUFF_COMP);i++){
508:         y[j+1].x=(double)c_pcmbuf[i]*(0.35875-0.48829*cos(2.0*PI*i/512)+0.14128*cos(4.0*P
I*i/512)-0.1168*cos(6.0*PI*i/512));
509:         y[j+1].y=(double)0.0;
510:         j++;
511:     }
512: }
```

```
513: void    rose(int i,int j){
514:     int k=i;
515:     for(;i<(k+BUFF_COMP);i++){
516:         y[j+1].x=(double)c_pcmbuf[i]*(0.762-cos(2.0*PI*i/512)+0.238*cos(4.0*PI*i/512))/2.
0;
517:         y[j+1].y=(double)0.0;
518:         j++;
519:     }
520: }
```

リスト3 FFT REC.C

```
 1: /*                  FFT 演算モジュール                   */
 2: /*  引数  入力   struct complex -> y[n]                */
 3: /*         出力   struct complex -> c[n]                */
 4: /*         データー数       int -> n                   */
 5: /*         正、逆のswitch    int -> iw                  */
 6:
 7:
 8: #include     <math.h>
 9:
10: typedef struct complex  CMPLX;
11: void    fft(CMPLX *,CMPLX *,int,int);
12:
13:
14: void     fft(CMPLX *c,CMPLX *y,int n,int iw){
15:    CMPLX str;
16:    double theta;
17:    int i,is,j,k,m,max;
18:
19:    for(i=1;i<=n;i++){
20:       c[i].x=y[i].x;
21:         c[i].y=y[i].y;
22:    }
23:
24:    for(i=j=1;i<=n;i++){
25:             if(i<j){
26:       str.x=c[j].x;
27:       c[j].x=c[i].x;
28:       c[i].x=str.x;
29:       str.y=c[j].y;
30:       c[j].y=c[i].y;
31:       c[i].y=str.y;
32:         }
33:       m=n/2;
34:
35:       while(j>m){
36:             j=j-m;
37:             m=m/2;
38:             if(m<2) break;
39:       }
40:       j=j+m;
41:     }
42:
43:    max=1;
44:
45:    while(max<n){
46:             is=max*2;
47:
48:       for(k=1;k<=max;k++){
49:      theta=PI*iw*(k-1)/max;
50:
51:       for(i=k;i<=n;i+=is){
52:          j=i+max;
53:          str.x=(c[j].x)*cos(theta)-(c[j].y)*sin(theta);
54:          str.y=(c[j].x)*sin(theta)+(c[j].y)*cos(theta);
55:          c[j].x=(c[i].x)-str.x;
56:          c[j].y=(c[i].y)-str.y;
57:          c[i].x=(c[i].x)+str.x;
58:          c[i].y=(c[i].y)+str.y;
59:       }
60:       }
61:       max=is;
62:    }
63:
64:    if(iw!=1){
65:       for(i=1;i<=n;i++){
66:       c[i].x=(c[i].x)/n;
67:       c[i].y=(c[i].y)/n;
68:       }
69:
70:    }
71: }
```

リスト4 WAVE.H

```
  1: #include       <iocslib.h>
  2: #include       <doslib.h>
  3: #include       <stdlib.h>
  4: #include       <stdio.h>
  5: #include       <conio.h>
  6: #include       <math.h>
  7:
  8: #define BUFSIZE 32                 /*ad pcm buff size*2 */
  9: #define PCMSIZE 1024               /*pcm buff size リアルタイム用*/
 10: #define BUFF_COMP  512
 11:
 12: typedef struct complex  CMPLX;
 13:
 14: struct  DMAREG{
 15:          unsigned      char      csr;
 16:          unsigned      char      cer;
 17:          unsigned      short     spare1;
 18:          unsigned      char      dcr;
 19:          unsigned      char      ocr;
 20:          unsigned      char      scr;
 21:          unsigned      char      ccr;
 22:          unsigned      short     spare2;
 23:          unsigned      short     mtc;
 24:          unsigned      char      *mar;
 25:          unsigned      long      spare3;
 26:          unsigned      char      *dar;
 27:          unsigned      short     spare4;
 28:          unsigned      short     btc;
 29:          unsigned      char      *bar;
 30:          unsigned      long      spare5;
 31:          unsigned      char      spare6;
 32:          unsigned      char      niv;
 33:          unsigned      char      spare7;
 34:          unsigned      char      eiv;
 35:          unsigned      char      spare8;
 36:          unsigned      char      mfc;
 37:          unsigned      short     spare9;
 38:          unsigned      char      spare10;
 39:          unsigned      char      cpr;
 40:          unsigned      short     spare11;
 41:          unsigned      char      spare12;
 42:          unsigned      char      dfc;
 43:          unsigned      long      spare13;
 44:          unsigned      short     spare14;
 45:          unsigned      char      spare15;
 46:          unsigned      char      bfc;
 47:          unsigned      long      spare16;
 48:          unsigned      char      spare17;
 49:          unsigned      char      gcr;
 50:          };
 51:
 52: /*ストレージ用のpcm-adpcm buffer address  の管理構造体*/
 53:    struct pcm_bank{
 54:       unsigned char  buf_name[22];
 55:       signed  short *pcmbuf_add; /*確保されたpcm用buf addressへのポインター*/
 56:       unsigned char  *adpcmbuf_add; /*確保されたadpcm用buf addressへのポインター*/
 57:       unsigned int   buf_len;  /*adpcm基準(adpcm data*4=pcmdata)のbuff長*/
 58:       struct  pcm_bank        *before;/*一つ前の構造体へのポインター*/
 59:       struct  pcm_bank        *next;  /*次の構造体へのポインター*/
 60:    };
 61:    typedef struct pcm_bank PCM_BANK;
 62:
 63:
 64: /*file検索された情報を格納する構造体*/
 65:
 66:    struct file_list{
 67:       unsigned char     file_name[22];/*得られたfilename*/
 68:       unsigned char     atr;
 69:       unsigned int      datetime;
 70:               size_t    len;
 71:       struct  file_list *before;
 72:       struct  file_list *next;
 73:    };
 74:    typedef struct file_list FILE_LIST;
 75:
 76:    void          adpcm_sample(unsigned char,unsigned char,unsigned char);
 77:    void          adpcm_start();
 78:    void          dma_setup();
 79:    char          dma_adpcm(unsigned char *,short);
 80:    void          dma_set_normal(unsigned char *,short);
 81:    void          dma_start();
 82:    void          dma_start2();
 83:    void          dma_stop();
 84:    void          clear_flag();
 85:    void          warikomi();
 86:    void          ret_vec();
 87:    void          draw_wave();
 88:    void          er_break();
 89:    void          draw_mod();
 90:    void          hd_copy();
 91:    void          init_screen();
 92:    void          end_screen();
 93:    void          init2_console();
 94:    void          gpalet_set();
 95:    void          scope_shatter(int,int);
 96:    void          rect_sw_off(short ,short ,int);
 97:    void          rect_sw_on(short ,short ,int );
 98:    void          mes_win(short ,short ,int );
 99:    void          mes_win2(short ,short ,int );
100:    void          gr_print(short ,short,unsigned char *,short ,int );
101:    void          glid1(int );
102:    void          glid2(int );
103:    void          waku1(int );
104:    void          panel_main(int );
105:    void          logo(int );
106:    void          tr_sw_off(short,short,short,int);
107:    void          tr_sw_on(short,short,short ,int);
108:    void          ad_to_pcm(signed short *,unsigned char *,int);
109:    void          waku2(int);
110:    void          calc_wave();
111:    void          tpalet_set();
112:    void          tpalet_ret();
113:    void          fft(CMPLX *,CMPLX *,int,int);
114:    void          gr_fft(int);
115:    void          calc_line(signed short *);/*pcm_lineは512ポイント分以上*/
116:    void          gr_clr(int);
117:    void          gr_data();
118:    void          rect(int,int);
119:    void          hann(int,int);
120:    void          blac(int,int);
121:    void          rose(int,int);
```

リスト5 DRAW_MOD.S

```
 1: *割り込み処理によりリアルタイムで _pcmbuf  の内容が書き換えられている状態で
 2: *draw_mod()では通常処理により _pcmbuf からデーターをエンドレスで読み
 3: *ドットによりグラフィック表示する
 4: *トリガーは+0のみ、読んでいる途中で書き換えが発生すると瞬間的に乱れるためﾀｲﾐﾝｸﾞ をとる
 5: *参照されるグローバル変数
 6: * _pcmbuf 16bit pcmdata の格納されているアドレス
 7: * _pcm_len pcmdata の個数
 8: * _last_pcm_add 割り込みで64 data 分のpcmbufの書き換えが終了した時点のアドレスポインター
 9: *1994 3/19
10:
11: .include         DOSCALL.equ
12:
13: GPAGE0_END       equ       $dffffe
14: GPAGE0           equ       $d80000
15: AUTOVAL          equ       -10
16: OFFSET_X         equ       -2
17: BUF_END          equ       -6
18: X_SIZE           equ       319
19: POINT_COL        equ       14         *green
20:
```

▶ あぁ，もっと技術力がほしい。勉強もしたいんだけど，超氷河期の就職活動中ではねぇ。
弧塚　一浩(21)栃木県

```
21:     .globl      _draw_mod
22:     .xref       _pcmbuf
23:     .xref       _pcm_len
24:     .xref       _last_pcm_add   *pcm buff address seek 保存ポインター
25:
26:     .text
27:     .even
28: _draw_mod:
29:     link        a6,#AUTOVAL
30:     movem.l     d3-d7/a3-a5,-(sp)
31:     move.w      #190,OFFSET_X(a6)       *x_offset=190
32:     moveq.l     #0,d6
33:     move.w      _pcm_len,d6
34:     sub.l       #320,d6         *pcmbuf_end-320  時間軸を倍にする時には 320*2=640
35:     moveq.l     #0,d3
36:     move.w      _pcm_len,d3
37:     sub.l       #340,d3         *pcmbuf_end-320       "         340*2=640
38:     move.l      d3,BUF_END(a6)  *pcmbuf_end-340
39:     moveq.l     #0,d3
40:     move.l      #_pcmbuf,d7
41:     bra         key_inp
42: seek_pcm:
43:     move.l      d3,d0
44:     add.l       d0,d0
45:     move.l      d7,a2
46:     lea         2(a2,d0.l),a1           *a1=(short)pcmbuf[i+1]
47:     move.l      d0,a0
48:     add.l       d7,a0                   *a0=(short)pcmbuf[i]
49: toriga_on:
50:     tst.w       (a0)
51:     bge         next_point
52:     tst.w       (a1)
53:     bge         new_point
54: next_point:
55:     addq.w      #2,a1
56:     addq.w      #2,a0           *pcmbuf[++]
57:     addq.l      #1,d3           *i++
58:     cmp.l       d3,d6           *i<max_buf
59:     bgt         toriga_on
60:     moveq.l     #0,d3
61: new_point:
62:     move.l      d3,d1
63:     add.l       #X_SIZE+1,d1
64:     move.l      d3,d4
65:     move.l      d3,d0
66:     add.l       d0,d0
67:     move.l      d0,a1
68:     add.l       d7,a1
69:
70: *pcm buffの書き換えと重なるおそれがある時はスキップ
71: *このルーチンを変える時は下の定数を調整すること
72:
73:     move.l      _last_pcm_add,a3
74: *       add.l #160,a3
75:     move.l      a1,a4
76:     add.l       #880,a4         *時間軸を倍にする時には #2000
77:     cmp.l       a1,a3
78:     ble         st
79:     cmp.l       a4,a3
80:     ble         seek_pcm
81: st:
82:     lea         _point_draw,a0
83:     bra         start_draw
84: calc_y:
85:     move.w      #255,a2
86:     sub.w       (a1)+,a2
87:     move.w      a2,(a0)+
88:     addq.l      #1,d4
89: start_draw:
90:     cmp.l       d4,d1
91:     bgt         calc_y
92:     move.w      OFFSET_X(a6),a2
93:     move.l      a2,AUTOVAL(a6)
94:     moveq.l     #0,d3
95:     lea         _point_del,a5
96:     lea         _point_draw,a4
97: gr_paint:
98:     move.w      (a5),d1
99:     move.w      -8(a6),d0
100:    bsr         gramadr
101:    move.w      #0,(a3)
102:    move.w      (a4),d1
103:    bsr         gramadr
104:    cmpa.l      #GPAGE0,a3              *clip
105:    blt         nopoint
106:    cmpa.l      #GPAGE0_END,a3          *clip
107:    bgt         nopoint
108:    move.w      #POINT_COL,(a3)
109: next:
110:    move.w      (a4)+,(a5)+
111:    addq.l      #1,AUTOVAL(a6)
112:    addq.l      #1,d3
113:    cmp.l       #X_SIZE,d3
114:    ble         gr_paint
115:    move.l      d4,d3
116:    cmp.l       BUF_END(a6),d3
117:    blt         key_inp
118:    moveq.l     #0,d3
119:
120: key_inp:
121:    moveq.l     #$74,d0
122:    trap        #15
123:    tst.w       d0
124:    beq         seek_pcm
125: *      move.w  #-1,-(sp)               *キーバッファクリア
126: *      DOS     KFLUSH                  *
127: *      addq.l  #2,sp                   *
128:    movem.l     (sp)+,d3-d7/a3-a5               *-42(a6)
129:    unlk        a6
130:    rts
131:
132: nopoint:
133:    move.w      #0,(a4)
134:    bra         next
135:
136: *      G-RAMアドレス関連モジュール
137: *      座標からG-RAMアドレスを求める
138: *      (d0.w,d1.w) → a3 = adr
139: *
140: gramadr:
141:    movem.l     d0-d1,-(sp)
142:    movea.w     d0,a3           *
143:    adda.w      d0,a3           *a0 = x*2
144:    moveq.l     #10,d0  *512*512
145:    ext.l       d1              *
146:    asl.l       d0,d1           *d1 = y*1024 (or y*2048)
147:    adda.l      d1,a3           *a0 = (x,y)と(0,0)のG-RAMアドレスの差
148:    adda.l      #GPAGE0,a3      *a0 = (x,y)のG-RAMアドレス GPAGE=0
149:    movem.l     (sp)+,d0-d1
150:    rts
151:
152:    .bss
153: _point_draw:
154:    .ds.b       640
155: _point_del:
156:    .ds.b       640
157:    .end
```

リスト6 WARIKOMI.S

```
1: *dma ad_pcmdata転送(継続動作モード)+リアルタイムad_pcm->pcm変換
2: *Cの関数として使用
3: *DMAはあらかじめ初期化しておくこと
4:
5:     .globl _warikomi
6:     .xdef  scaleval
7:     .xdef  levelchg
8: *
9: *以下のラベルは C のｸﾞﾛｰﾊﾞﾙ変数として用意されていること
10: *また念のため最適化を避けるため volatileにしておく
11: *使用する時にはメインでスーパーバイザーモードに入っておく
12: *adpcmよりサンプリング、ad_pcmbuf1,2に交互に継続転送
13: *更にad_pcmbuf1,2よりpcmbufにﾘｱﾙﾀｲﾑpcm変換(short size)される
14: *タイマ関係の割り込みはMFPにより禁止する Cメインで
15:
16:     .xref       _dma                    *dmac channel3 address pointer
17:     .xref       _inp_frag               *継続動作用buff選択フラグ
18:     .xref       _ad_pcmbuf1             *top ad_pcm buff1 address
19:     .xref       _ad_pcmbuf2             *top ad_pcm buff2 address
20:     .xref       _pcmbuf                 *top pcmbuff address
21:     .xref       _len                    *ad_pcmbuff 長 固定
22:     .xref       _len_pcm_buf    *pcm buff 長 のカウンタ last_pcm_addからpcmbufの終わりまでの長
さ
23:     .xref       _pcm_len                *original pcmbuff 長(固定)
24:     .xref       _last_pcm_add   *pcm buff address seek 保存ポインター
25:     .xref       _last_val               *一つ前ののpcmdata を保存
26:     .xref       _val_x                  *一つ前のscaleを 保存する変数
27:     .xref       _vol_sift               *pcm データーの倍率(シフト数)
28:
29:     *dma register channel 3
30:
31:     .offset             0
32:     csr:                .ds.b   1
33:     cer:                .ds.b   1
34:     spare1:             .ds.w   1
35:     dcr:                .ds.b   1
36:     ocr:                .ds.b   1
37:     scr:                .ds.b   1
38:     ccreg:              .ds.b   1
39:     spare2:             .ds.w   1
40:     mtc:                .ds.w   1
41:     mar:                .ds.l   1
42:     spare3:             .ds.l   1
43:     dar:                .ds.l   1
44:     spare4:             .ds.w   1
45:     btc:                .ds.w   1
46:     bar:                .ds.l   1
47:     spare5:             .ds.l   1
48:     spare6:             .ds.b   1
49:     niv:                .ds.b   1
50:     spare7:             .ds.b   1
51:     eiv:                .ds.b   1
52:     spare8:             .ds.b   1
53:     mfc:                .ds.b   1
54:     spare9:             .ds.w   1
55:     spare10:            .ds.b   1
56:     cpr:                .ds.b   1
57:     spare11:            .ds.w   1
58:     spare12:            .ds.b   1
59:     dfc:                .ds.b   1
60:     spare13:            .ds.l   1
61:     spare14:            .ds.w   1
62:     spare15:            .ds.b   1
63:     bfc:                .ds.b   1
64:     spare16:            .ds.l   1
65:     spare17:            .ds.b   1
66:     gcr:                .ds.b   1
67:
68:
69:     .text
70:
71: * a2->_dma address
72: * d0->local val
73: *_last_pcm_add ->次に転送されるpcm buff アドレス
74:
75: _warikomi:
76:     movem.l     d0-d7/a0-a6,-(sp)
77:     move.l      _last_pcm_add,a1  *PCMbufのaddress既に更新されている
78:     lea.l       _vol_sift,a6
79:     move.b      _inp_frag,d0
80:     bne         _sab_b
81:     move.l      _dma,a2
82:     move.b      (a2),d0                 *csr to d0
83:     or.b        #%0100_0000,d0          *csr-btcビットクリア
84:     move.b      d0,(a2)
85:     lea.l       _ad_pcmbuf1,a0
86:     move.l      a0,bar(a2)
87:     moveq.l     #0,d0
88:     move.w      _len,d0         *変換データ個数
89:     move.w      d0,btc(a2)
90:     move.b      #$5,bfc(a2)
91:     move.b      ccreg(a2),d5
92:     or.b        #%0100_0000,d5
93:     bsr         _just_adpcm_to_pcm              *変換処理  間に合わなければ cer->err
94:     move.b      d5,ccreg(a2)    *ccr->cntビット  クリア 次の転送へ
95:     move.b      #1,_inp_frag
96:     bra         _end
97:
98:     *ad_pcmbuf2 への転送準備
99: _sab_b:
100:    move.l      _dma,a2
101:    move.b      (a2),d0
102:    or.b        #%0100_0000,d0
103:    move.b      d0,(a2)
104:    lea.l       _ad_pcmbuf2,a0
105:    move.l      a0,bar(a2)
106:    moveq.l     #0,d0
107:    move.w      _len,d0
108:    move.w      d0,btc(a2)
109:    move.b      #$5,bfc(a2)
110:    move.b      ccreg(a2),d5
111:    or.b        #%0100_0000,d5
112:    bsr         _just_adpcm_to_pcm
113:    move.b      d5,ccreg(a2)
```

▶パソコン(X68000 ACE)が壊れました。でもまたX68000を買います。なぜなんだろう…
…かっぱえびせんみたいな要素があるなX68000には。　　　九重　至宏(23)京都府

```
114:    move.b      #0,_inp_frag
115:
116: _end:
117:    movem.l     (sp)+,d0-d7/a0-a6
118:    rte
119:
120:
121:
122: _just_adpcm_to_pcm:                            *ADPCM→PCM変換
123:
124:    * < a0=adpcm data buffer
125:    * < a1=pcm data buffer
126:    * < d0.l=data lenghth
127:
128:    lea         scaleval(pc),a5
129:    lea         levelchg(pc),a4
130:    moveq.l     #0,d3
131:    move.w      _val_x,d7
132:    moveq.l     #$0f,d4
133:    add.l       d0,d0
134: __atp_lp:
135:    move.b      (a0),d1
136:    and.w       d4,d1
137:    tst.b       d4
138:    bpl         __neg_d4
139:    lsr.b       #4,d1           *get 4bit data
140:    addq.l      #1,a0
141: __neg_d4:
142:    not.b       d4
143:    bsr         calc_pcm_val    *実際の計算
144:    move.b      (a6),d3
145:    asr.w       d3,d2                                   *倍率
146:    move.w      d2,(a1)+        *move pcm data to buffer
147:    subq.l      #1,d0           *adpcmbufのsize=32まで
148:    bne         __atp_lp
149:    move.w      _len_pcm_buf,d6  *_len_pcm_bufは_lenの倍数
150:    sub.w       #64,d6          *32data adpcm ->64data pcm
151:    beq         _new            *128byte むだになる！
152:    move.w      d6,_len_pcm_buf  *_len_pcm_bufへd6を保存
153:    move.l      a1,_last_pcm_add
154:    move.w      d7,_val_x
155:    rts
156: _new:
157:    lea.l       _pcmbuf,a3      *seek pointer をpcmbuffの先頭へ戻す
158:    move.l      a3,_last_pcm_add
159:    move.w      _pcm_len,d6
160:    move.w      d6,_len_pcm_buf
161:    move.w      d7,_val_x
162:    rts
163:
164: calc_pcm_val:
165:    * < d1.b=adpcm value
166:    * < d7.w=scale level
167:    * > d2.w=pcm value
168:    * > d7.w=next scale level
169:    * > d1.b=adpcm*2
170:    * X d3 d2
171:    add.b       d7,d7
172:    move.w      (a5,d7.w),d3    *=d
173:    lsr.b       d7
174:    moveq.l     #0,d2           *=def
175:    btst        #2,d1
176:    beq         tst_bit1
177:    move.w      d3,d2
178: tst_bit1:
179:    lsr.w       d3
180:    btst        #1,d1
181:    beq         tst_bit0
182:    add.w       d3,d2
183: tst_bit0:
184:    lsr.w       d3
185:    btst        #0,d1
186:    beq         decide_def
187:    add.w       d3,d2
188: decide_def:
189:    lsr.w       d3
190:    add.w       d3,d2
191:
192:    btst        #3,d1           *minus or plus??
193:    beq         plus_lastval
194:    neg.w       d2              *case not zero
195: plus_lastval:
196:    add.w       _last_val,d2
197: *       bsr     chk_ovf
198:    move.w      d2,_last_val    *d2=pcmdata
199:
200:    add.b       d1,d1
201:    add.w       (a4,d1.w),d7            *scalelevl+=levelchg(adpcm value)
202:    bmi         rst_sclv
203:    cmpi.w      #48,d7
204:    bls         allend
205:    moveq.l     #48,d7
206: allend:
207:    rts
208: rst_sclv:
209:    moveq.l     #0,d7
210:    rts
211:
212:    .data
213:    .even
214:
215: scaleval:
216:    dc.w        16,17,19,21,23,25,28
217:    dc.w        31,34,37,41,45,50,55
218:    dc.w        60,66,73,80,88,97,107
219:    dc.w        118,130,143,157,173,190,209
220:    dc.w        230,253,279,307,337,371,408
221:    dc.w        449,494,544,598,658,724,796
222:    dc.w        875,963,1060,1166,1282,1411,1552
223: levelchg:
224:    dc.w        -1,-1,-1,-1,2,4,6,8
225:    dc.w        -1,-1,-1,-1,2,4,6,8
226:
```

リスト7 AD_TO_PCM.S

```
  1: *adpcm->pcm変換ストレージ用
  2: *
  3: *引き数        short           pcmbuff
  4: *               char            adpcmbuff
  5: *               int             adpcmbuff長
  6: *
  7: .xdef   _ad_to_pcm
  8: .xref   scaleval
  9: .xref   levelchg
 10:
 11: .offset 4
 12: pcmbuf_add              ds.l    1       *pcmbbufへのポインター
 13: adpcmbuf_add            ds.l    1       *adpcmbufへのポインター
 14: buf_len                 ds.l    1       *adpcmbufの大きさ バイト
 15:
 16: .text
 17:
 18: SAVESIZE        equu            10*4 *register待避
 19:
 20: _ad_to_pcm:                 *ピッチチェンジやレベルチェンジを
 21:            *行わない単なるADPCM→PCM変換
 22:    * < a0=adpcm data buffer
 23:    * < a1=pcm data buffer
 24:    * < d0.l=data size
 25:    movem.l     a0-a3/d0-d5,-(sp)
 26:    move.l      pcmbuf_add+SAVESIZE(sp),a1
 27:    move.l      adpcmbuf_add+SAVESIZE(sp),a0
 28:    move.l      buf_len+SAVESIZE(sp),d0
 29:
 30:    lea         scaleval(pc),a2
 31:    lea         levelchg(pc),a3
 32:    moveq.l     #0,d3
 33:    moveq.l     #0,d5
 34:    moveq.l     #$0f,d4
 35:    add.l       d0,d0
 36:    clr.w       last_val
 37: __atp_lp:
 38:    move.b      (a0),d1
 39:    and.w       d4,d1
 40:    tst.b       d4
 41:    bpl         __neg_d4
 42:    lsr.b       #4,d1           *get 4bit data
 43:    addq.w      #1,a0
 44: __neg_d4:
 45:    not.b       d4
 46:    bsr         calc_pcm_val    *実際の計算
 47:
 48:    move.w      d2,(a1)+        *add pcm data to buffer
 49:
 50:    subq.l      #1,d0
 51:    bne         __atp_lp
 52:    movem.l(sp)+,a0-a3/d0-d5
 53:
 54:    rts
 55:
 56: calc_pcm_val:
 57:    * < d1.b=adpcm value
 58:    * < d5.w=scale level
 59:    * > d2.w=pcm value
 60:    * > d5.w=next scale level
 61:    * > d1.b=adpcm*2
 62:    * X d3 d2
 63:    add.b       d5,d5
 64:    move.w      (a2,d5.w),d3    *=d
 65:    lsr.b       d5
 66:
 67:    moveq.l     #0,d2           *=def
 68:
 69:    btst        #2,d1
 70:    beq         tst_bit1
 71:    move.w      d3,d2
 72: tst_bit1:
 73:    lsr.w       d3
 74:    btst        #1,d1
 75:    beq         tst_bit0
 76:    add.w       d3,d2
 77: tst_bit0:
 78:    lsr.w       d3
 79:    btst        #0,d1
 80:    beq         decide_def
 81:    add.w       d3,d2
 82: decide_def:
 83:    lsr.w       d3
 84:    add.w       d3,d2
 85:
 86:    btst        #3,d1           *minus or plus??
 87:    beq         plus_lastval
 88:    neg.w       d2              *case not zero
 89: plus_lastval:
 90:    add.w       last_val(pc),d2
 91: *       bsr     chk_ovf
 92:    move.w      d2,last_val     *d2=pcmdata
 93:
 94:    add.b       d1,d1
 95:    add.w       (a3,d1.w),d5            *scalelevl+=levelchg(adpcm value)
 96:    bmi         rst_sclv
 97:    cmpi.w      #48,d5
 98:    bls         allend
 99:    moveq.l     #48,d5
100: allend:
101:    rts
102: rst_sclv:
103:    moveq.l     #0,d5
104:    rts
105:
106:    .data
107:    .even
108:
109:    .bss
110: last_val:       ds.w    1
```

リスト8 Makefile

```
  1: CC=GCC -c -O -Q -w #-z-heap = 40000
  2: AS= has
  3: ALAGS=-u -w0
  4: LK=hlk
  5: LLIB= c:¥lib¥clib.l c:¥lib¥baslib.l c:¥lib¥doslib.l c:¥lib¥iocslib.l c:¥lib¥floatfnc.l c
:¥lib¥gnulib.l c:¥lib¥proflib.l
  6: LFLAGS=
  7: SFILE= WARIKOMI.S DRAW_MOD.S AD_TO_PCM.S
  8: CFILE= OCR.C DRAW_WAVE.C FFT_REC.C #DRAW_MOD.C
  9: COBJ=$(CFILE:C=O)                    # 構成オブジェクト
 10: SOBJ=$(SFILE:S=O)
 11:
 12: OCR.X: $(COBJ) $(SOBJ)
 13:        $(LK) $(LFLAGS)  $(COBJ) $(SOBJ)  $(LLIB)
 14:
 15: %.O: %.S
 16:        $(AS) $(ALAGS) $<
 17: %.O: %.C
 18:        $(CC) $(CFLAGS) $<
```

▶つ，ついにOh!Xにプログラムが紹介されました。バンザーイ！　8月号のカラーページをめくっていたら，見慣れた画面が目に飛び込んできました。その瞬間，友だちと世界一決定戦(テストプレイともいう)を繰り広げた日々が懐かしく思い出されました。

今村　哲矢(22)東京都

こちらシステムX探偵事務所

FILE-XXVI

illustration：T. Takahashi

生命の遺伝システムを模倣する

Shibata Atsushi 柴田 淳

今回は以前に考えた遺伝のシステムについて見直していきます。そして，そのときに問題となっていた突然変異率のジレンマを回避するために，実際の遺伝でも起こるといわれる，「交叉」をサンプルプログラムに取り入れてみました。

柴田淳(以下Ats)：最近，圧縮に凝っているんですよ。

マスター(以下M)：圧縮っていうと，あの布団を厚手のビニールに入れて掃除機で空気を吸い出す……。

Ats：違いますよ。ファイルの容量を縮めるほうの圧縮です。

琴張春香(以下春)：なによ2人とも，掛け合い漫才の教科書みたいな会話して。

琴張護(以下護)：これで柴田君のボケ突っ込みが入れば完璧でした。

M：ところで，そのファイル圧縮がどうしたんですか？　まさか，新しい圧縮アルゴリズムを作ろうとかいうんじゃないでしょうね。

Ats：凝っているといっても，そうじゃなくて，とりあえずいろいろな参考書を引っ張り出して，いろんな圧縮方法を組み合わせたり，何種類かの特徴のあるデータを食わせてみて，圧縮率の変わり方を調べる程度ですけどね。

春：ふーん。そんなことを面白がる人の神経って，わたしわからないわ。

M：だいいち，その参考書とやらに載っている方法というのは，きっと確立された圧縮アルゴリズムなんだろうから，そこそこ効率のいい圧縮ができるんでしょう。

春：そうよね。百歩譲って，より効率のいい圧縮アルゴリズムを見つけることは面白いとするわよ。だけど，他人の作った結果のわかっているプログラムを試してみるのが，そんなに価値のあることなのかしら。

Ats：ぼくも最初はそう思っていたんです。ところが，実際に本に載っているようなアルゴリズムを試してみると，LHAのような有名な圧縮ソフトほど高い圧縮率は得られないんです。

護：確か，著名な圧縮ソフトはほとんど，複数のアルゴリズムを組み合わせて高い圧縮率を実現していたのではなかったでしょうか。

M：なるほど。それでいろいろなアルゴリズムを組み合わせてみるわけですね。

Ats：圧縮されたものを展開すると圧縮前の状態に戻るタイプ，俗にいう可逆圧縮っていうのは，基本的に2種類の方法で圧縮できるんです。ひとつは，データの繰り返し部分を見つけて圧縮する方法。もうひとつは，ファイル中のデータの使用頻度によって圧縮する方法です。

春：じゃあ，この2種類を組み合わせればうまくいくんじゃないかしら。

護：しかし，最初の方法でデータの繰り返し部分を減らすと，圧縮後のデータの使用頻度がどれも似通った数値になってしまいます。すると，2番目の方法を用いたときの圧縮率が激減してしまうのです。

春：なるほど。ファイル圧縮のアルゴリズムにはそういうジレンマがあるのね。

遺伝子進化の中立説

M：そういえば，遺伝するアルゴリズムを作ったときも，同じようなことをいってませんでしたっけ？

護：確か，遺伝子の突然変異率を上げると，バラエティに富んだ個体が生まれるが集団が安定しない。逆に変異率を下げると，安定はするが個体同士がどれも似通ってしまう，というようなことではなかったでしょうか。

Ats：そうなんですよ。せっかく計算モデルを固定しないですむようになったのに，そのことがネックになって，どうしてもうまくいかないんですよね。で，いろいろな本を調べてみたんです。

M：で，なにかわかったんですか？

Ats：まず，いままでの手法は，まったくランダムに遺伝子を突然変異させていましたよね。

春：遺伝子をランダムに変異させて，偶然出来上がった優秀な遺伝子だけを育てていくっていうことだったわよね。

Ats：ところが，それだけではだめらしいということがわかったんです。

M：え，ちょっと待ってください。生物が進化してきた過程でも，ランダムに起こる遺伝子の複製エラーが突然変異を生み出して，より環境に適応したものが生き残ってきたわけでしょう。

護：そうです。現実に起こっていることを模倣してうまくいかないというのは，話が矛盾しているように思えるのですが。

Ats：いや，生物もまったくランダムな遺伝子の変異だけで進化してきたわけではないらしいんですよ。というか，遺伝子が偶然に突然変異するだけでは，多様な環境に巧みに適応したいまのような生物は生まれえないといったほうが正しいかな。

M：いまいち納得いかないですね。

Ats：じゃあ逆にこういったらどうでしょう。遺伝するアルゴリズムを作ったとき，遺伝子の変異率を上げすぎると結果がばらつき，逆に低くすると変化が起こらない，というジレンマがありましたが，これと同じことが，現実の遺伝子にも起こりうるんじゃないでしょうか？

M：それは確かにそうですけど……。

Ats：分子生物学という研究分野があるんですが，この分野の大きな成果のひとつに「中立説」というものがあるんです。進化を引き起こす突然変異は，自然淘汰に対して中立である，という学説なんです。

春：もう少しわかりやすくいってよ。

護：つまり，突然変異によって，環境によりよく適応した生物が「生まれやすくなる」わけではないということですね。

Ats：いやそれどころか，この中立説を提唱している木村資生氏は「中立説では，(淘汰に)有利な突然変異はごくまれにしか起

こらないので……無視してもかまわない」とまでいっています。

春：でも現実を見ると，いろんな形をした生物がいるわけよね。

M：しかも，それぞれ環境に適応して生きている。これはどういうわけですか。

Ats：ひとついえるのは，地球の歴史上，何度か環境が激変した時期があり，それが生命進化を加速するうえでかなり重要な役割を果たしているらしいということです。

護：環境の激変が進化にどう寄与しているというのですか。

Ats：中生代から新生代への移り変わりに，それまで地上をほとんど支配していた恐竜が絶滅しましたよね。すると，いままで恐竜が独占していた生命資源が，一気に過剰状態になるわけです。

M：なるほど。すると，それまでは影を潜めていた哺乳類が一気に繁殖できるようになるわけですね。

Ats：つまり，環境が激変すると，淘汰の制約が急に緩やかになるんです。その間に生物は多様性を獲得して，広まっていくというわけですね。

M：だとすると，一般に信じられている進化論のイメージというのは，かなりおおざっぱなものということになりますね。

巧妙な遺伝システム

護：ところで，環境の激変が進化を早めるとはいえ，依然として突然変異率のジレンマは残っているのではないでしょうか？

Ats：確かにそうですね。でも，現実の遺伝のメカニズムには，このジレンマを回避する仕組みがいくつか組み込まれているんです。

M：どんなのがあるんですか。

Ats：２つの遺伝子同士でお互いの遺伝子の要素を交換する「交叉」という現象があるんです。この交叉はたいてい２つの遺伝子の同じ要素同士で起こるんですが，たまに似たような部分が入れ替わることがあるんです。

護：似たような部分が入れ替わると，遺伝子は長くなったり短くなったりしますね。

Ats：で，そうやって遺伝子本来の役割であるタンパク質のコーディングには関係のない「非コード領域」と呼ばれる部分が遺伝子に蓄積していくんです。驚いたことに，人間の遺伝子ではこの非コード領域は98％にもなるそうなんです。

春：つまり，タンパク質の合成に必要な部分はたった２％ということ？

M：その非コード領域っていうのはなんのためにあるんですか。

Ats：それがよくわからないらしいんです。ただ，以前に獲得した環境適応に有利な部分を，遺伝子の中に保っておくという働きは考えられますね。

護：すると，環境が変わって淘汰の制約が緩まったときに，非コード領域がコード領域として復活するということが起こりえますね。

Ats：そうなんです。つまりこの非コード領域は生物が進化の過程で多様性を獲得するための仕組みのひとつといえそうですね。

春：そのほかには？

Ats：脊椎動物のような高等な生物の遺伝子は，２倍体といって２つの遺伝子が対になって存在しています。それぞれにタンパク質の情報があるわけですが，同じ遺伝子の組と違う遺伝子の組では，出来上がるタンパク質に差が出てくるんです。

M：優性遺伝子と劣性遺伝子というやつですね。

Ats：たとえば，人間の血液型でいうなら，ＡＢ型とＯ型の両親からは，Ａ型かＢ型の子供しか生まれません。

護：Ｏ型はＡ型に対してもＢ型に対しても劣性であるため，表面に現れないというわけですね。

Ats：この遺伝子の優劣を利用して，有利な変異だけを蓄積することができます。そのほかにも，ラディング鎖とリーディング鎖の変異率の違いとか，交叉の際もまとまって遺伝する超遺伝子とか，有利な変異を蓄積する仕組みはたくさん見られます。このあたりの細かいことは参考文献に詳しく書いてあります。

遺伝システムの模倣

Ats：とはいっても，これらをすべてプログラムに取り入れるのは大変です。そこで，とりあえず遺伝子の交叉と，それによって起こる遺伝子重複を実現する方法を考えてみましょう。

M：確かいままでの方法では，遺伝子は置き換えられるか，いちばん最後に新しい要素が加えられるかのどちらかでしたよね。そうやって，遺伝子を突然変異させてたんじゃありませんでしたっけ。

護：つまり，突然変異は，要素の入れ替えと，遺伝子の冗長化という２つの操作に分けられるわけですね。

Ats：今回は，遺伝子を長くする操作をするとき，最後につけ足すのではなく遺伝子の間に割り込ませるようにするわけです。遺伝子は，自分自身に次の要素のポインタをもつリスト構造で表現されています。最後に要素をつけ足すには，最後の要素のポインタに次の要素をつけ足せばいいんです

図１　新たな遺伝子要素を間に挿入する場合

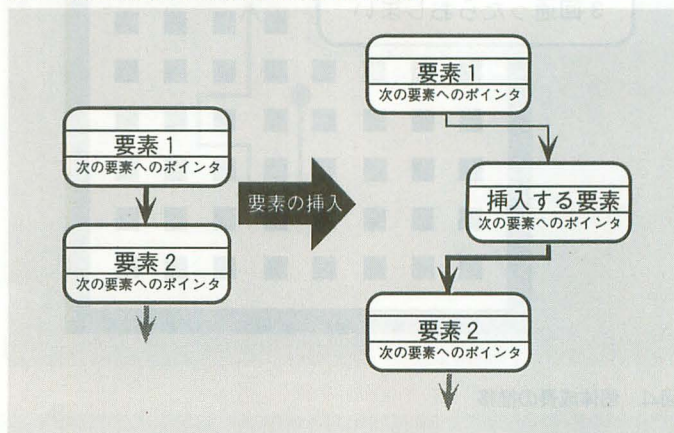
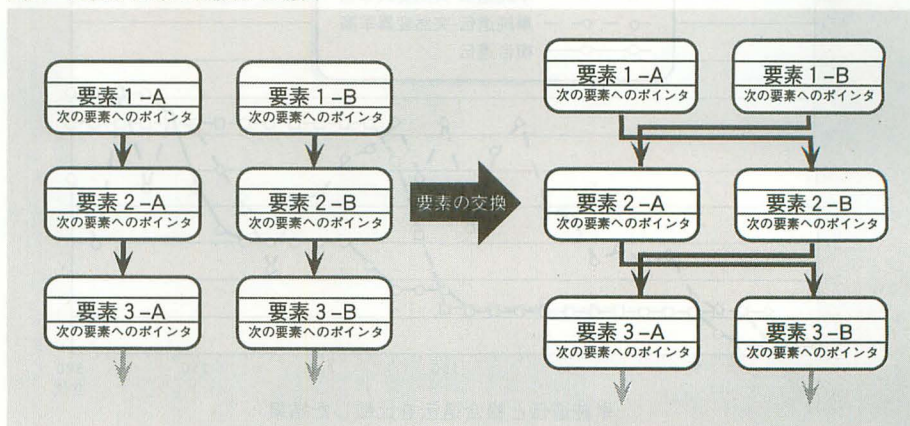



図２　遺伝子要素を交換する場合

が，途中に割り込ませるとなると，ちょっとややこしくなります。
M：というと？
Ats：要するに，図1のように挿入する直前の要素のポインタと，挿入する要素のポインタの2つを書き換えなければならないんです。また，遺伝子の交叉を実現するためには2つの遺伝子の要素を交換しなければならないわけですが，その場合はもっとややこしくて……。
M：なるほど。図2のように4つの要素のポインタを書き換えなければならないんですね。
Ats：では次に，実際に遺伝アルゴリズムをもつプログラムを作ってみて，生命の遺伝システムを模倣することによってどんな効果が現れるか検証してみましょう。
春：今回はどんなのを作るの？
Ats：ええと，図3にあるようなルールで，格子状の迷路の中を，遺伝子に従って個体の体を伸ばしていきます。なるべく長く体を伸ばした個体が適応して生き残っていきます。遺伝子の要素は，先月琴張さんの作ったサブルーチン群を使って，メモリに動的に割り当てます。
護：それは光栄です。
Ats：あのサブルーチンがあるおかげで，遺伝子の操作がたいへん簡単になりましたよ。たとえば，169行目からの関数Mutateで突然変異を起こしているんですが，適応しない個体を消去するために必要な処理は，204行から208行だけなんです。消去とは逆の遺伝子をコピーするのも簡単ですしね。
護：223行から228行までがその処理ですね。
Ats：あと，先ほど図1，2で説明した遺伝子要素の交換と挿入，つまり交叉と遺伝子重複ですが，実際の処理は321行目からの関数Exchangeと，290行目からの関数Duplicateで行っています。
春：関数Exchangeが交叉で，関数Duplicateが遺伝子重複の処理になるのね。
M：リストでは，次の遺伝子要素を示すポインタを書き換えていますね。図と見比べるとわかりやすい。
Ats：さて，こうやって適応度の高い個体を増やしながら，決められた確率で突然変異，交叉，遺伝子重複などを繰り返しながら個体は育っていくわけです。その結果を，図4のグラフに示してみました。折れ線グラフの横軸は世代数で，縦軸は個体全体の体の長さの平均を表しています。
春：図4にあるもう2つの折れ線は？
Ats：これは比較のために，以前までの方法で個体を成長させたときの結果を出力したものです。突然変異だけを扱った遺伝なので，単純遺伝と名づけてみました。今回の方法は，単純に対して複合遺伝と呼んでみましょう。
護：なるほど。突然変異率を高くすると，個体の体の長さは上下に激しく振れて，なかなか安定しないようです。
M：逆に突然変異率を低くすると，個体の育ちが悪くなる。
春：でもグラフを見ると，今回の方法より，突然変異率を低くした単純遺伝のほうが，早く成長しているように見えるけど。
Ats：それには理由があるんです。今回のようなルールで個体を成長させるとき，遺伝子の長さが個体の身長を決めるのに大きな要素になります。で，前回までの方法だと，遺伝子は後ろにつけ足されますから，比較的早く遺伝子長が長くなるんです。
春：だけど，今回の方法でも遺伝子を重複させるんだから，遺伝子は長くなるんじゃない。
Ats：いや，それが違うんです。遺伝子重複によって遺伝子は長くなるけど，長くなった部分は非コード領域として蓄積するので，結局遺伝子長そのものには影響を与えないんです。
M：最初のほうでそんなことをいってましたね。
Ats：しかし，非コード領域に突然変異が起こると，いままで眠っていた遺伝子が活動を始めることがあります。そうやって徐々に適応度の高い個体が生まれていくわけです。
春：そういえば確かに，今回の方法のグラフはほかのと比べると安定したカーブになっているわね。
Ats：あと，今回は試しませんでしたが，定期的に淘汰の制約を緩めてやると，もう少し安定した進化が見られるようです。

（つづく）

図3 サンプルの個体成長のルール

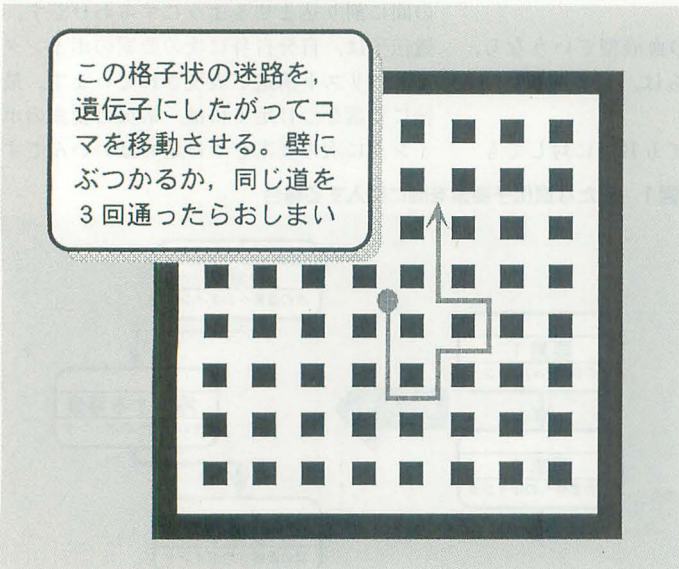
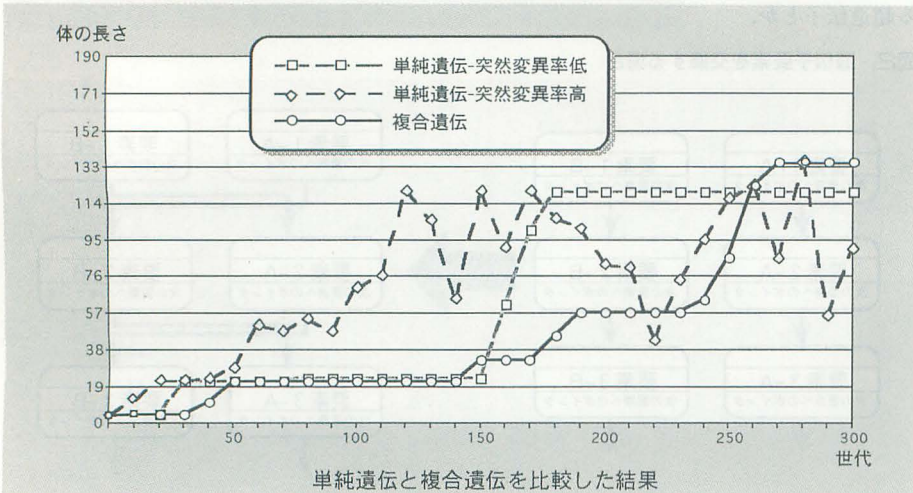



図4 個体成長の推移

単純遺伝と複合遺伝を比較した結果

参考文献
和田健之介，岩波科学ライブラリ「デジタル生命の進化」，岩波書店
神原武志ほか，講談社ブルーバックス「パソコンで探る生命科学シミュレーション」，講談社
宮田隆，講談社ブルーバックス「分子進化学への招待」，講談社

リスト

```
  1: /*
  2:           生物の遺伝システムの模倣 任意のキーで終了
  3:           (注) 8月号掲載のリストUtilities.cが必要
  4: */
  5:
  6: #include <stdio.h>
  7: #include <stdlib.h>
  8: #include <stddef.h>
  9: #include <string.h>
 10:
 11: #define _GetParam(ptr)  ((*(ObjectListPtr)ptr).params)
 12: #define _GetSig(ptr)    ((*(ObjectListPtr)ptr).object_sig)
 13:
 14: #include "Utilities.c"
 15: /*8月号掲載のリストをインクルード*/
 16: typedef struct  {
 17:         void    *actionFunc,*get_paramFunc,
 18:                 /*実行関数, パラメータ取得関数のポインタ*/
 19:                 *next_action;
 20:                 /*次の遺伝子要素のポインタ*/
 21:         long    direction,x,y,seats,condition,no_cond;
 22:                 /*パラメータ群*/
 23: } Actions,*ActionsPtr;
 24: /* 遺伝子要素の構造体 */
 25: typedef struct  {
 26:         void*   next_act;
 27:         long    seats,cond,no_con;
 28: } Param,*ParamPtr;
 29: /* 遺伝子要素初期化用の構造体*/
 30: typedef struct  {
 31:         long    x,y,dir,depth,times,ret;
 32: } ActionPar,*ActionParPtr;
 33: /* 実行関数に渡す構造体。点の位置や方向などを記憶*/
 34: typedef struct  {
 35:         long    par1,par2,par3,depth;
 36: } GPParam,*GPParamPtr;
 37: /* パラメータ取得用の構造体 */
 38: typedef long    actions(void*,ActionParPtr);
 39: typedef long    get_param(void*,long depth,GPParamPtr);
 40:
 41: enum    {
 42:         junction = 1000,
 43:         action };
 44: /* オブジェクトのID */
 45: enum    {
 46:         go_straight = 0,
 47:         turn_right,
 48:         turn_left };
 49:
 50: #define dir_max 4
 51: long    dir_x[dir_max] = {
 52:         0,1,0,-1 },
 53:         dir_y[dir_max] = {
 54:         -1,0,1,0 };
 55:
 56: #define map_width       11
 57: #define map_height      11
 58: #define finish_x        9
 59: #define finish_y        9
 60: #define collide         3
 61: #define max_times       200
 62: #define max_stump       3
 63: #define max_ind         200
 64:
 65: long    map[map_width][map_height] =
 66:         { 1,1,1,1,1,1,1,1,1,1,1,
 67:           1,0,0,0,0,0,0,0,0,0,1,
 68:           1,0,1,0,1,0,1,0,1,0,1,
 69:           1,0,0,0,0,0,0,0,0,0,1,
 70:           1,0,1,0,1,0,1,0,1,0,1,
 71:           1,0,0,0,0,0,0,0,0,0,1,
 72:           1,0,1,0,1,0,1,0,1,0,1,
 73:           1,0,0,0,0,0,0,0,0,0,1,
 74:           1,0,1,0,1,0,1,0,1,0,1,
 75:           1,0,0,0,0,0,0,0,0,0,1,
 76:           1,1,1,1,1,1,1,1,1,1,1 },
 77:           tmap[map_width][map_height];
 78: /* 地図 */
 79: void    Initialize();
 80: void    grow();
 81: void    Mutate();
 82: void    ClearTMap(void);
 83: long    AddInd(ObjectListPtr ob);
 84: void    Change(ObjectListPtr ob);
 85: void    Duplicate(ObjectListPtr ob);
 86: void    Exchange(ObjectListPtr ob1,ObjectListPtr ob2);
 87: ObjectListPtr   GetAction(ObjectListPtr,long);
 88: void    CreateAction(void* this,void* param);
 89: void    DistructAction(void* this);
 90: void*   CopyAction(void* this);
 91: void    CreateJunction(void* this,void* param);
 92: void    DistructJunction(void* this);
 93: void*   CopyJunction(void* this);
 94: long    aAction(void*,ActionParPtr);
 95: long    aGetParam(void*,long,GPParamPtr);
 96: long    jAction(void*,ActionParPtr);
 97: long    jGetParam(void*,long,GPParamPtr);
 98: long    GetMapValue(long,long);
 99:
100: ObjectListPtr   IndList[max_ind];
101: /* 個体のリスト */
102: long    total = 1,wx,wy,ox,oy,wpos,
103:                 foots[max_ind],
104:                 dep[max_ind],IndCount;
105:
106: void Initialize(void)
107: /* 初期化用の関数 */
108: {
109: long    i;
110:         allmem();
111:         for( i = 0; i != 200; i++ ) {
112:                 IndList[i] = NULL;
113:                 foots[i] = 0;
114:                 dep[i] = 0;
115:         }
116: }
117:
118: void    main()
119: {
120: long    siz = sizeof(Actions),gen = 0;
121: Param   tmpP;
122:
123:         Initialize();
124: /* オブジェクトを登録 */
125:         RegisterObject(CreateJunction,DistructJunction,
126:                         CopyJunction,siz,junction);
127:         RegisterObject(CreateAction,DistructAction,
128:                         CopyAction,siz,action);
129:         tmpP.next_act = NULL;
130:         tmpP.cond = go_straight+1;
131:         IndList[0] = ConstructObject(action,&tmpP);
132:         do {
133:                 gen++;
134:                 printf( "世代-%d:",gen);
135:                 grow();
136:                 Mutate();
137:         }while( !kbhit() );
138:         return;
139: }
140:
141: void    grow()
142: /* オブジェクトを成長させる */
143: {
144: long    i;
145: ActionPar       ap;
146: ActionsPtr      tmp;
147: GPParam gp;
148:         IndCount = 0;
149:         for( i = 0; i != max_ind; i++ ) {
150:                 if( IndList[i] != NULL ) {
151:                         ap.x = 5;
152:                         ap.y = 5;
153:                         ap.dir = 2;
154:                         ap.times = 0;
155:                         ap.ret = 0;
156:                         ClearTMap();
157:                         tmp = (ActionsPtr)((*IndList[i]).p
arams);
158:                         while( !((actions*)(*tmp).actionFu
nc)(IndList[i],&ap) );
159:                                 foots[i] = ap.times;
160:                         tmp = (ActionsPtr)((*IndList[i]).p
arams);
161:                         gp.depth = 0;
162:                         ((get_param*)(*tmp).get_paramFunc)
(IndList[i],9999,&gp);
163:                         dep[i] = gp.depth;
164:                         IndCount++;
165:                 }
166:         }
167: }
168:
169: void    Mutate()
170: /* 突然変異+α */
171: {
172: long    i,j,t,tTime,tDepth,cnt = 0;
173: ObjectListPtr   tmp;
174:         /* まず, 評価値の高い順に並べ替える */
175:         for( i = 0; i != max_ind; i++ ) {
176:                 for( j = 0; j != max_ind-1; j++ ) {
177:                         if( IndList[j] != NULL &&
178:                                 IndList[j+1] != NULL &&
179:                                 foots[j] < foots[j+1] ) {
180:                                 t = foots[j];
181:                                 foots[j] = foots[j+1];
182:                                 foots[j+1] = t;
183:                                 t = dep[j];
184:                                 dep[j] = dep[j+1];
185:                                 dep[j+1] = t;
186:                                 tmp = IndList[j];
187:                                 IndList[j] = IndList[j+1];
188:                                 IndList[j+1] = tmp;
189:                         }
190:                 }
191:         }
192:         tTime = 0;
193:         tDepth = 0;
194:         for( i = 0; IndList[i] != NULL && i < 5; i++ ) {
195:                 tTime += foots[i];
196:                 tDepth += dep[i];
197:                 cnt++;
198:         }
199:         printf( "体長の平均%3d:遺伝子長の平均%3d",
200:                                 tTime/cnt,tDepth/cnt);
201:         printf("¥n");
202:         if( IndCount > 195 ) {
203:                 /* 適応度の低い遺伝子を殺す */
204:                 t = max_ind-1-(IndCount-195);
205:                 for( i = t; i!= IndCount; i++ ) {
206:                         DeleteObject(IndList[i]);
207:                         IndList[i] = NULL;
208:                 }
```

```
209:                    t = IndCount-195-1;
210:                    if( t <= 0 ) {
211:                            t = 1;
212:                    }
213:            } else {
214:                    t = max_ind-IndCount-1;
215:                    if( t > IndCount ) {
216:                            t = IndCount;
217:                            if( t <= 0 ) {
218:                                    t = 1;
219:                            }
220:                    }
221:            }
222:            /* 適応度の高い遺伝子をコピー */
223:            for( i = 0; i != t; i++ ) {
224:                    tmp = CopyObject(IndList[i]);
225:                    if( AddInd(tmp) ) {
226:                            DeleteObject(tmp);
227:                    }
228:            }
229:            /* 突然変異+α */
230:            for( i = 0; IndList[i+1] != NULL; i++ ) {
231:                    if( rand() < RAND_MAX/20 ) {
232:                            Change(IndList[i]);
233:                    }
234:                    if( rand() < RAND_MAX/11 ) {
235:                            Duplicate(IndList[i]);
236:                    }
237:                    if( rand() < RAND_MAX/10 ) {
238:                            Exchange(IndList[i],IndList[i+1]);
239:                    }
240:            }
241:
242: }
243:
244: long    AddInd(ObjectListPtr ob)
245: /* オブジェクトを加える */
246: {
247: long    i = 0;
248:         while( IndList[i] != NULL && i < max_ind ) {
249:                 i++;
250:         }
251:         if( i == max_ind ) {
252:                 return( 1 );
253:         }
254:         IndList[i] = ob;
255:         return( 0 );
256: }
257:
258: void    Change(ObjectListPtr ob)
259: /* 突然変異 */
260: {
261: long    i;
262: GPParam gp;
263: ObjectListPtr   op1,op2,op3;
264: ActionsPtr      tmpA;
265: Param   tmpP;
266:         tmpA = (ActionsPtr)((*ob).params);
267:         gp.depth = 0;
268:         ((get_param*)(*tmpA).get_paramFunc)(ob,9999,&gp);
269:         i = gp.depth;
270:         if( i < 2 ) {
271:                 return;
272:         }
273:         i = (double)rand()/RAND_MAX*(i-2)+1;
274:         op1 = GetAction(ob,i-1);
275:         op2 = GetAction(ob,i);
276:         op3 = GetAction(ob,i+1);
277:         (*(ActionsPtr)((*op2).params)).next_action = NULL;
278:         DeleteObject(op2);
279:         tmpP.next_act = op3;
280:         tmpP.cond = 0;
281:         tmpP.no_con = -1;
282:         if( rand() > RAND_MAX/2 ) {
283:                 op2 = ConstructObject(action,&tmpP);
284:         } else {
285:                 op2 = ConstructObject(junction,&tmpP);
286:         }
287:         (*(ActionsPtr)((*op1).params)).next_action = op2;
288: }
289:
290: void    Duplicate(ObjectListPtr ob)
291: /* 遺伝子重複 */
292: {
293: long    i,j;
294: GPParam gp;
295: ObjectListPtr   op1,op2,op3;
296: ActionsPtr      tmpA;
297: Param   tmpP;
298:         tmpA = (ActionsPtr)((*ob).params);
299:         gp.depth = 0;
300:         ((get_param*)(*tmpA).get_paramFunc)(ob,9999,&gp);
301:         i = gp.depth;
302:         j = (double)rand()/RAND_MAX*(i);
303:         op1 = GetAction(ob,j);
304:         if( j+1 < i ) {
305:                 op3 = GetAction(ob,i+1);
306:         } else {
307:                 op3 = NULL;
308:         }
309:         tmpP.next_act = op3;
310:         tmpP.cond = -1;
311:         tmpP.no_con = 1;
312:         if( rand() > RAND_MAX/2 ) {
313:                 op2 = ConstructObject(action,&tmpP);
314:         } else {
315:                 op2 = ConstructObject(junction,&tmpP);
316:         }
317:         (*(ActionsPtr)((*op1).params)).next_action = op2;
318:
319: }
320:
321: void    Exchange(ObjectListPtr ob1,ObjectListPtr ob2)
322: /* 単純交叉 */
323: {
324: long    i;
325: GPParam gp;
326: ObjectListPtr   opA1,opA2,opA3,opB1,opB2,opB3;
327: ActionsPtr      tmpA,tmpB;
328:         if( ob1 == NULL || ob2 == NULL ) {
329:                 return;
330:         }
331:         tmpA = (ActionsPtr)((*ob1).params);
332:         gp.depth = 0;
333:         ((get_param*)(*tmpA).get_paramFunc)(ob1,9999,&gp);
334:         i = gp.depth;
335:         tmpB = (ActionsPtr)((*ob2).params);
336:         ((get_param*)(*tmpB).get_paramFunc)(ob2,9999,&gp);
337:         if( i > gp.depth ) {
338:                 i = gp.depth;
339:         }
340:         i = (double)rand()/RAND_MAX*(i-1)+1;
341:         opA1 = GetAction(ob1,i-1);
342:         opA2 = GetAction(ob1,i);
343:         opA3 = GetAction(ob1,i+1);
344:         opB1 = GetAction(ob2,i-1);
345:         opB2 = GetAction(ob2,i);
346:         opB3 = GetAction(ob2,i+1);
347:         if( opA1 == NULL || opB1 == NULL ) {
348:                 return;
349:         }
350:         (*(ActionsPtr)((*opA1).params)).next_action = opB2
;
351:         if( opA2 != NULL ) {
352:                 (*(ActionsPtr)((*opA2).params)).next_actio
n = opA3;
353:         }
354:         (*(ActionsPtr)((*opB1).params)).next_action = opA2
;
355:         if( opB2 != NULL ) {
356:                 (*(ActionsPtr)((*opB2).params)).next_actio
n = opB3;
357:         }
358: }
359:
360: ObjectListPtr   GetAction(ObjectListPtr op,long depth)
361: {
362: ObjectListPtr   tmpA;
363:         tmpA = op;
364:         while( depth > 0 ) {
365:                 tmpA = (*(ActionsPtr)((*tmpA).params)).nex
t_action;
366:                 if( tmpA == NULL ) {
367:                         break;
368:                 }
369:                 depth--;
370:         }
371:         return( tmpA );
372: }
373:
374: void    ClearTMap()
375: {
376: long i,j;
377:         for( i = 0; i != map_width; i++ ) {
378:                 for( j = 0; j != map_height; j++ ) {
379:                         tmap[i][j] = 0;
380:                 }
381:         }
382: }
383: /* オブジェクト action 用の関数群 */
384: void    CreateAction(void* this,void* param)
385: {
386:         (*(ActionsPtr)_GetParam(this)).actionFunc =
387:                 aAction;
388:         (*(ActionsPtr)_GetParam(this)).condition =
389:                 ((double)rand()/(RAND_MAX+1))*3;
390:         if( (*(ParamPtr)param).cond ) {
391:                 (*(ActionsPtr)_GetParam(this)).condition =
392:                         (*(ParamPtr)param).cond-1;
393:         }
394:         (*(ActionsPtr)_GetParam(this)).get_paramFunc =
395:                 aGetParam;
396:         if( (*(ParamPtr)param).next_act != NULL ) {
397:             (*(ActionsPtr)_GetParam(this)).next_action =
398:                         (*(ParamPtr)param).next_act;
399:         }
400: }
401:
402: void    DistructAction(void* this)
403: {
404:         if( (*(ActionsPtr)_GetParam(this)).next_action !=
NULL ) {
405:                 DeleteObject((*(ActionsPtr)_GetParam(this)
).next_action);
406:         }
407:         StandardDistructor(this);
408: }
409:
410: void*   CopyAction(void* this)
411: {
412: ObjectListPtr   newObject;
413:         newObject = StandardCopier(this);
414:         if( (*(ActionsPtr)_GetParam(this)).next_action !=
NULL ) {
415:                 (*(ActionsPtr)(*newObject).params).next_ac
tion =
```

▶エプソンのノートパソコン用SIMMにはシャープのRAMが使われているのがある。8個もシャープのチップが並ぶ様はなかなかカッコイイ。　久米　豊信(27)群馬県

```
416:                    CopyObject((*(ActionsPtr)_GetParam(this))
.next_action);
417:            }
418:            return( newObject );
419: }
420: /* オブジェクト junction 用の関数群 */
421: void    CreateJunction(void* this,void* param)
422: {
423: long    tmp;
424:            (*(ActionsPtr)_GetParam(this)).actionFunc =
425:                    jAction;
426:            (*(ActionsPtr)_GetParam(this)).get_paramFunc =
427:                    jGetParam;
428:            tmp = -(double)rand()/(RAND_MAX-1)*40-20;
429:            if( rand() > RAND_MAX/2 ) {
430:                    (*(ActionsPtr)_GetParam(this)).no_cond = 1
;
431:            } else {
432:                    (*(ActionsPtr)_GetParam(this)).no_cond = 0
;
433:            }
434:            (*(ActionsPtr)_GetParam(this)).condition = tmp;
435:            if( (*(ParamPtr)param).cond ) {
436:                    (*(ActionsPtr)_GetParam(this)).condition =
437:                            (*(ParamPtr)param).cond;
438:            }
439:            if( (*(ParamPtr)param).no_con != -1 ) {
440:                    (*(ActionsPtr)_GetParam(this)).no_cond =
441:                            (*(ParamPtr)param).no_con;
442:            }
443:            if( (*(ParamPtr)param).next_act != NULL ) {
444:                    (*(ActionsPtr)_GetParam(this)).next_action
=
445:                            (*(ParamPtr)param).next_act;
446:            }
447: }
448:
449: void    DistructJunction(void* this)
450: {
451:            if( (*(ActionsPtr)_GetParam(this)).next_action !=
NULL ) {
452:                    DeleteObject((*(ActionsPtr)_GetParam(this)
).next_action);
453:            }
454:            StandardDistructor(this);
455: }
456:
457: void*   CopyJunction(void* this)
458: {
459: ObjectListPtr   newObject;
460:            newObject = StandardCopier(this);
461:            if( (*(ActionsPtr)_GetParam(this)).next_action !=
NULL ) {
462:                    (*(ActionsPtr)(*newObject).params).next_ac
tion =
463:                     CopyObject((*(ActionsPtr)_GetParam(this))
.next_action);
464:            }
465:            return( newObject );
466: }
467:
468: long    aAction(void* this,ActionParPtr   parm)
469: {
470: ActionsPtr      tmp;
471: long    tx,ty,ret = 0;
472:
473:            tmp = (*(ActionsPtr)_GetParam(this)).next_action;
474:            if( (*parm).ret > 0 ) {
475:                    (*parm).ret--;
476:                    ((actions*)(*(ActionsPtr)_GetParam(tmp)).a
ctionFunc)(tmp,parm);
477:            }
478:            switch((*(ActionsPtr)_GetParam(this)).condition) {
479:                    case go_straight :
480:                            tx = (*parm).x+dir_x[(*parm).dir];
481:                            ty = (*parm).y+dir_y[(*parm).dir];
482:                            if( !GetMapValue(tx,ty) ) {
483:                                    (*parm).times++;
484:                                    (*parm).x = tx;
485:                                    (*parm).y = ty;
486:                                    tmap[tx][ty]++;
487:                                    if( tmap[tx][ty] >= max_st
ump ) {
488:                                            return( collide );
489:                                    }
490:                            } else {
491:                                    return( collide );
492:                            }
493:                            break;
494:                    case turn_right :
495:                            (*parm).dir++;
496:                            (*parm).dir %= 4;
497:                            break;
498:                    case turn_left :
499:                            (*parm).dir += 3;
500:                            (*parm).dir %= 4;
501:                            break;
502:            }
503:            (*parm).depth++;
504:            if( tmp ) {
505:                    ret = ((actions*)
506:                    (*(ActionsPtr)_GetParam(tmp)).actionFunc)(
tmp,parm);
507:            }
508:            if( (*parm).ret < 0 ) {
509:                    (*parm).ret++;
510:            }
511:            if( (*parm).ret == 0 && tmp ) {
```

```
512:                    ret = ((actions*)(*(ActionsPtr)_GetParam(t
mp)).actionFunc)(tmp,parm);
513:            }
514:            return( ret );
515: }
516:
517: long    aGetParam(void* this,long depth,GPParamPtr gp)
518: {
519: ActionsPtr tmp;
520:            if( depth == 0 ) {
521:                    (*gp).par1 =
522:                    (*(ActionsPtr)_GetParam(this)).condition;
523:                    (*gp).par2 = action;
524:                    return( 0 );
525:            }
526:            depth--;
527:            (*gp).depth++;
528:            tmp = (*(ActionsPtr)_GetParam(this)).next_action;
529:            if( tmp ) {
530:                    ((get_param*)(*(ActionsPtr)_GetParam(tmp))
.get_paramFunc)(tmp,depth,gp);
531:            }
532:            return( 0 );
533: }
534:
535: long    jAction(void* this,ActionParPtr parm)
536: {
537: long    val = 0,ret = 0,t;
538: ActionsPtr      tmp;
539:            tmp = (*(ActionsPtr)_GetParam(this)).next_action;
540:            if( (*parm).ret > 0 ) {
541:                    (*parm).ret--;
542:                    ((actions*)
543:                    (*(ActionsPtr)_GetParam(tmp)).actionFunc)
544:                                    (tmp,parm);
545:            }
546:            switch( (*parm).dir ) {
547:            case 0 :
548:                    val = GetMapValue((*parm).x,(*parm).y-1);
549:                    break;
550:            case 1 :
551:                    val = GetMapValue((*parm).x+1,(*parm).y);
552:                    break;
553:            case 2 :
554:                    val = GetMapValue((*parm).x,(*parm).y+1);
555:                    break;
556:            case 3 :
557:                    val = GetMapValue((*parm).x-1,(*parm).y);
558:                    break;
559:            }
560:            t = (*(ActionsPtr)_GetParam(this)).condition;
561:            if( (*(ActionsPtr)_GetParam(this)).no_cond ) {
562:                    val ^= 1;
563:            }
564:            if( val ) {
565:                    (*parm).ret = t;
566:                    if( t <= 0 ) {
567:                            return(0);
568:                    }
569:            }
570:            if( tmp ) {
571:                    ret = ((actions*)
572:                    (*(ActionsPtr)_GetParam(tmp)).actionFunc)
573:                                    (tmp,parm);
574:            }
575:            if( (*parm).ret < 0 ) {
576:                    (*parm).ret++;
577:            }
578:            if( (*parm).ret == 0 && tmp ) {
579:                    ret = ((actions*)
580:                    (*(ActionsPtr)_GetParam(tmp)).actionFunc)
581:                                    (tmp,parm);
582:            } else {
583:                    ret = 0;
584:            }
585:            return( ret );
586: }
587:
588: long    jGetParam(void* this,long depth,GPParamPtr gp)
589: {
590: ActionsPtr tmp;
591:            if( depth == 0 ) {
592:                    (*gp).par1 =
593:                    (*(ActionsPtr)_GetParam(this)).condition;
594:                    (*gp).par2 = junction;
595:                    (*gp).par3 =
596:                    (*(ActionsPtr)_GetParam(this)).no_cond;
597:                    return( 0 );
598:            }
599:            depth--;
600:            (*gp).depth++;
601:            tmp = (*(ActionsPtr)_GetParam(this)).next_action;
602:            if( tmp ) {
603:                    ((get_param*)(*(ActionsPtr)_GetParam(tmp))
.get_paramFunc)(tmp,depth,gp);
604:            }
605:            return( 0 );
606: }
607:
608: long    GetMapValue(long x,long y)
609: {
610:            if( x >= map_width || y >= map_height || x < 0 ||
y < 0 ) {
611:                    return( 1 );
612:            }
613:            return( map[y][x] );
614: }
```

▶秋葉原でメガドライブ用の「ゆみみみっくす」を見つけたので，つい買ってしまった。これでいつかはCD-ROMドライブを買わなければいけなくなった。どんなゲームかも知らないのに……。
石井　英一郎(21)千葉県

愛読者プレゼント

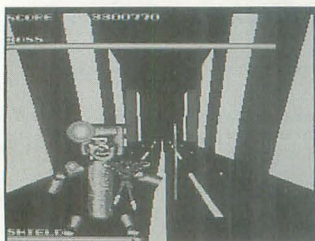
プレゼントの応募方法　とじ込みのアンケートハガキの該当項目をすべてご記入のうえ，希望するプレゼント番号をハガキ右下のスペースにひとつ記入してお申し込みください。締め切りは1995年9月18日の到着分までとします。当選者の発表は1995年11月号で行います。また，雑誌公正競争規約の定めにより，当選された方はこの号のほかの懸賞に当選できない場合がありますので，ご了承ください。

同人ソフトセットI

くろわっさん
X68000用　5"2HD版　1,200円(税込)

CLISS　3名
X68000用　5"2HD版　500円(税込)

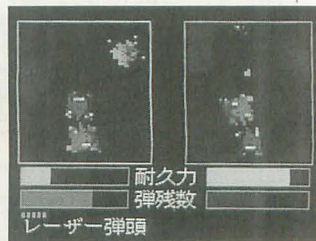
TAKERU事務局　☎052(824)2493

パロディ3Dシューティングとキャラクターがかわいい風船アクションゲームをセットで遊ぼう。

同人ソフトセットII

GUARDIAN・RS
X68000用　5"2HD版　800円(税込)

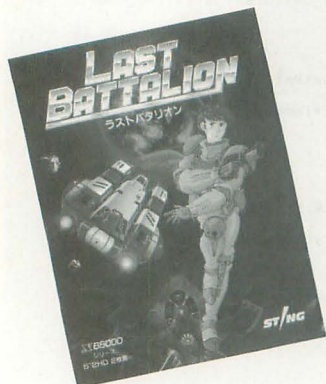
情け無用Fire!2　3名
X68000用　5"2HD版　1,300円(税込)

TAKERU事務局　☎052(824)2493

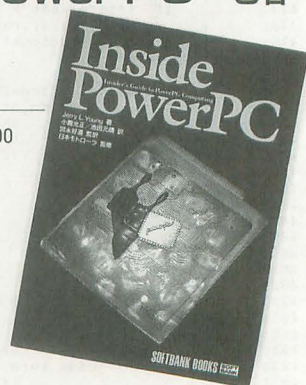
両ソフトともグリグリ回転する背景がウリのシューティングゲーム。あんまり遊びすぎて目を回さないようにね。

ラストバタリオン

X68000用　5"2HD版　8,800円(税別)
3名

スティング

PC-Engineからパワーアップ移植された，溜め撃ちショットがハデハデのシューティングゲーム。惑星ウラノスを救うために撃って撃って撃ちまくるのだ！

Inside Power PC　3名

ソフトバンク　☎03(5642)8100

最近話題のRISCチップ，Power PCの内部を網羅した解説本です。最新の技術に触れたい方にお勧め。

7月号プレゼント当選者

1 SUPER LIBRARY SERIES　A)YELLOW MAGIC ORCHESTRA/YMO　(千葉県)小川　貴也　B)SOLID STATE SURVIVOR/YMO　(東京都)川田　洋　2 スターモビール　(新潟県)小川　比佐夫　3 スライス　(神奈川県)安井　繁明　4 キーパー(東京都)藍原　和久　5 A)THE World of X68000　(東京都)松島　竜　B)THE World of X68000II　(栃木県)大嶋　靖浩
以上の方々が当選しました。商品は順次発送いたしますが，入荷状況などにより遅れる場合もあります。

THE SENTINEL

〈対応機種一覧〉●MZ-80 K/C/700/1500 ●MZ-80 B/2000 ●MZ-2500/2861 ●X1 ●X1 turbo/Z ●PC-8001/8801/88 ●SMC-777/C ●PASOPIA/5 ●PASOPIA/7 ●FM-7/77/AV ●MSX/2/2+/turbo R ●PC-286/386/486/9801/98/9821 ●X68000/X68030
掲載されたプログラムの利用には各機種用のS-OS"SWORD"システムが必要です。

第160部 FE ラインプリントルーチン詳細
第161部 MISSILE SYSTEM

●**ゲームにおけるキー操作**

以前にも紹介したことがあるゲームにおけるキー操作。最近曖昧になってきているというツッコミがあったことですし，ここである程度ガイドラインを定義してしまいましょう。

1) ゲーム中，いつでもSHIFT+BREAKでモニタに戻れるようにする

2) ゲームにおける移動キーは，テンキー(2,4,6,8)，カーソルキー，Kを中心としたキー(M,J,L,I)のすべてをサポートする。そして，なるべくなら小文字も同時に判別するのが望ましい

3) ごく普通のシューティングゲームであれば，ショットは連射を基本とする(S-OSでは，同時キー入力ができないため，ショットを撃ちながら移動するということができないため)。撃ちっぱなしというのも問題なので，トグルスイッチによってショットのON/OFFを切り替えるようにする（トリガはZ周辺，スペースキーを使用。パズルゲームであれば5，Kもサポート）

もちろん，ゲームによってはぜんぜん操作体系が違う場合もあります。適宜ゲームに合わせてアレンジしましょう。

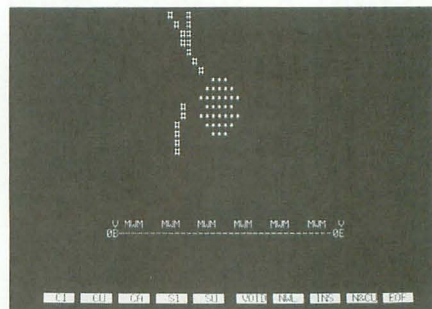
●**移動方向判別法**

ところで，複数のタイプの移動キーを判別するのは，なかなか面倒な作業です。ここで，どのようにプログラミングすればいいのか解説します。

まず，上下左右方向を0～3のコードに割り当てます。そして，判別するキーのキャラクタコードテーブルに対応するように，移動方向の数値テーブルを作成すればいいのです。たとえば，5を中心としたテンキーとKを中心としたキーを使うならば，

```
CHR_TBL:
    DM  "8246IMJL"
DIR_NUM:
    DB  0,1,2,3,0,1,2,3
```

以上のようなテーブルを作ります。そして，入力されたキーのキャラクタコードが，何番目で判別されたかカウントします。次にDIR_NUM:のテーブルから，そのカウンタに対応する値を取り出せばお望みの移動方向を得られるというものです。"M"キーであれば，CHR_TBL:の5番目にありますから，最終的には下方向を示す"1"の数値を得られます。

ところが，わざわざ移動方向の数値テーブルを用意しなくても，目的の数値を得る方法があります。テーブルを見ればわかるとおり，移動方向の数値テーブルは，きれいに0,1,2,3を繰り返しています。これがどういう意味をもつかというと，何番目でキャラクタが判別されたかというカウンタの下位3ビットが，移動方向の数値テーブルと同じなのです。

つまり，わざわざテーブルを用意してやる必要がなく，カウンタをAレジスタに設定して，

```
AND  3
```

で移動方向が得られてしまうのです。ちなみに，8方向であれば，

```
AND  7
```

とすればいいのはわかりますよね。もちろん，この方法が使えるのは，移動方向が2の倍数であることが条件です。判別する移動方向が2の倍数でない場合は，素直に移動方向テーブルを用意するか，むりやり2の倍数になるようにテーブルの中にダミーデータを付加する必要があります。

だいたいこのような感じでしょうか。基本的にアセンブラであろうとコンパイラであろうとも原理は変わりません。これからS-OSでゲームを作ろうとしている人は，参考にしてください。

1995■インデックス

全機種共通
S-OS"SWORD"要

FEver.1.0

ラインプリントルーチン詳細

Sakamaki Katsumi
坂巻 克巳

お約束のFEver.1.0のラインプリントルーチンの詳細です。これで，各機種専用のラインプリントルーチンを簡単に制作できるでしょう。また，バグ情報もありますので修正をしてください。

さあさあ，寄ってらっしゃい見てらっしゃい，FEラインプリントルーチンの詳細だよ！　これを読めばもうばっちり。あなたもFE用の各機種専用ラインプリントルーチンが組めてしまうという代物だあ！　なんてたいそうなものでもありませんが，お約束の情報です。他人のプログラムを眺めるのは，久しぶり。できるかどうか，解析を始めるまでちょっとだけ緊張しちゃいましたが，まあなんとかなりました。さあ，がんばって説明していきましょうか。

概要説明

まず，7月号に掲載した全機種共通ラインプリントルーチン(リスト5)とX1/turbo専用ルーチン(リスト6)を見比べてください。ひととおり見るとすぐにわかると思いますが，両リストともに結構似ています(当たり前か)。違うのは，全機種共通ルーチンでは，

```
CALL _PRINT
```

となっているところが，専用ルーチンでは，なにやらI/Oポートに書き込みを行っている点と，ラベルlocate_init以降の内容が異なっている点です。

つまり，表示位置設定のサブルーチンであるlocate_initを，各機種専用の画面アドレス計算ルーチンに差し替え，さらにCALL _PRINTの代わりに各機種専用の1文字描画ルーチンを埋め込めば，完成となります(その際には，とりあえずレジスタを全部保存しておきましょう)。

これだけ理解していれば，各機種専用のラインプリントルーチンを制作できるでしょう。

深く探ってみる

以上の情報だけでも各機種専用のラインプリントルーチンが制作できますが，それだけでは記事として成り立ちません。もうちょっと，ラインプリントルーチン周りの動作原理を探ってみることにします。

再掲載したリスト1の全機種共通ラインプリントルーチンを見てください。まず，8～26行のラベル定義部分の内容を説明しましょう。

```
 9 _PRINT     EQU  1FF4H
10 _LOC       EQU  201EH
```

この2行は問題ありませんね。S-OS用の1文字表示ルーチンとカーソル位置設定用のサブルーチンです。

```
13 next_chr   EQU  3069H
```

このルーチンはFE本体にあるサブルーチンで結構重要なものです。とりあえず，名前のとおり次にある文字を取り出すルーチンだと解釈しておきましょう。詳しくは，リスト全体を把握したあとに説明します。

```
15 console_x      EQU  4591H
16 console_y      EQU  4592H
17 console_width  EQU  4593H
18 console_length EQU  4594H
```

以上はFE内のワークエリアアドレスを定義しています。名前のとおり表示座標関係のものです。上から順番に，現在いるカーソルのX座標，カーソルのY座標，カーソルの動ける範囲(横)，そして最後は……解読していません。実際，サブルーチン内で使用されていないものなので，ラインプリントルーチン内では無視してかまわないと判断してしまいました。

```
20 line_data_adr  EQU  45D2H
21 line_print_y   EQU  45D1H
22 view_offset    EQU  45A9H
23 tab_info_table EQU  45FCH
```

上から順番に，表示する1行分のデータが格納されている先頭アドレス，画面の一番上のY座標(かな？　ちょっと自信ないけど)，横スクロールのオフセット座標，タブ情報テーブルの先頭アドレスを格納しているアドレスです。

そして，33行からいよいよプログラムが始まります。34～37行は説明するまでもありませんね。レジスタの保存を行っているだけです。で，問題なのが次にある1行です。

```
38 LD   E,(IX)
```

解析するときに最後まで困ったのがこのIXレジスタの役割でした。とりあえず，Eレジスタの役割がわかれば問題なしと判断し，IXレジスタは深く追求していません(つまりわからなかったのだ)。さっさとリストを眺めていきましょう。

```
39 CALL locate_init
```

先ほども説明したとおり，表示すべきラインのカーソル位置を指定するサブルーチンをコールしています。

```
40 LD   A,(nm_chr_atri)
41 LD   (chr_atri),A
```

通常キャラクタのアトリビュート(色情報)を取り出し，ワークに格納しています。

```
42 LD   HL,(line_data_adr)
43 LD   IY,tab_info_table
```

HLレジスタに表示する1行データが格納されているアドレスをセットし，そして，IYレジスタにタブ情報テーブルの先頭アドレスをセットしています。

以上がいわゆる初期化部分ですね。

次にプログラムのメイン部分に移ります。45~51行は，横スクロールのオフセット分だけ，1行データのポインタであるHLレジスタを進めています。そして，53~61行で実際の表示を行っています。横幅に応じて表示回数を変えているところがポイントですね。

なお，以上のプログラムでは，Dレジスタをカウンタに使用していることがわかれば，簡単に理解できるでしょう。

最後は，レジスタ内容を復帰させてリターンしているだけです。

next_chrは?

これで全体の構造がだいたいわかりました。あとは，説明を先送りにしていた，next_chrというサブルーチンを理解できれば完璧です。といってもちょっとだけ処理が複雑だったので，いちばん解析に時間がかかったのがこのサブルーチン。などどいう個人的な感想はともかく，リスト2を見ながらさっさと説明してしまいましょう。

簡単にいうとnext_chrでは，バッファからキャラクタの取り出し，そして特殊文字の判別とそれに応じたアトリビュートの設定を行っています。引数と返り値は，

[引数]

Eレジスタ:通常モード(0)，タブモード(1)，改行モード(2)の判別フラグ

HLレジスタ:表示しようとしている文字列バッファのポインタ

IYレジスタ:タブ情報テーブルのポインタ

[返り値]

Aレジスタ:表示するキャラクタコード

chr_atri:表示するキャラクタの色情報

以上のようになっています。

そして，ここでの押さえておきたいポイ

リスト1

```
0000                 1  ;-----------------------------------
0000                 2  ;
0000                 3  ;  line_print routine for FE v1.04
0000                 4  ;  for general
0000                 5  ;
0000                 6  ;-----------------------------------
0000                 7
0000                 8
1FF4 P               9  _PRINT  EQU     1FF4H
201E P              10  _LOC    EQU     201EH
0000                11
0000                12
3069 P              13  next_chr        EQU     3069H
0000                14
4591 P              15  console_x       EQU     4591H
4592 P              16  console_y       EQU     4592H
4593 P              17  console_width   EQU     4593H
4594 P              18  console_length  EQU     4594H
0000                19
45D2 P              20  line_data_adr   EQU     45D2H
45D1 P              21  line_print_y    EQU     45D1H
45A9 P              22  view_offset     EQU     45A9H
45FC P              23  tab_info_table  EQU     45FCH
0000                24
3016 P              25  nm_chr_atri     EQU     3016H
45D4 P              26  chr_atri        EQU     45D4H
0000                27
0000                28
0000                29          OFFSET  0C000H-4400H
4400                30          ORG     4400H
4400                31
4400                32
4400                33  line_print:
4400 C5             34          PUSH    BC
4401 D5             35          PUSH    DE
4402 E5             36          PUSH    HL
4403 FD E5          37          PUSH    IY
4405 DD 5E 00       38          LD      E,(IX)
4408 CD 3C 44       39          CALL    locate_init
440B 3A 16 30       40          LD      A,(nm_chr_atri)
440E 32 D4 45       41          LD      (chr_atri),A
4411 2A D2 45       42          LD      HL,(line_data_adr)
4414 FD 21 FC 45    43          LD      IY,tab_info_table
4418                44  ;
4418 3A A9 45       45          LD      A,(view_offset)
441B 57             46          LD      D,A
441C 14             47          INC     D
441D 15             48  lp1:    DEC     D
441E CA 27 44       49          JP      Z,lp2
4421 CD 69 30       50          CALL    next_chr
4424 C3 1D 44       51          JP      lp1
4427                52  ;
4427 3A 93 45       53  lp2:    LD      A,(console_width)
442A 57             54          LD      D,A
442B 15             55          DEC     D
442C CD 69 30       56  lp3:    CALL    next_chr
442F                57  ;
442F CD F4 1F       58          CALL    _PRINT
4432                59  ;
4432 15             60          DEC     D
4433 C2 2C 44       61          JP      NZ,lp3
4436 FD E1          62          POP     IY
4438 E1             63          POP     HL
4439 D1             64          POP     DE
443A C1             65          POP     BC
443B C9             66          RET
443C                67
443C                68
443C                69  locate_init:
443C 3A D1 45       70          LD      A,(line_print_y)
443F 67             71          LD      H,A
4440 3A 92 45       72          LD      A,(console_y)
4443 84             73          ADD     A,H
4444 3A 91 45       74          LD      A,(console_x)
4447 6F             75          LD      L,A
4448                76  ;
4448 CD 1E 20       77          CALL    _LOC
444B C9             78          RET
444C                79
444C                80
```

リスト2

```
3069               100  ;---------------------------------------
3069               101  ; Subroutine for line_print
3069               102
3069               103
3069               104  next_chr:
3069 7B            105          LD      A,E
306A B7            106          OR      A
306B C2 C2 30      107          JP      NZ,nxt_c1
306E 7E            108  nxt_c4: LD      A,(HL)
306F 23            109          INC     HL
3070 FD 23         110          INC     IY
3072 FE 20         111          CP      32
3074 D0            112          RET     NC
3075 FE 09         113          CP      __TAB
3077 CA 92 30      114          JP      Z,nxt_c5
307A FE 0D         115          CP      __CR
307C CA 9E 30      116          JP      Z,nxt_c6
307F B7            117          OR      A
3080 CA AA 30      118          JP      Z,nxt_c7
3083 FE 1A         119          CP      __EOF
3085 CA B6 30      120          JP      Z,nxt_c8
3088 3A 13 30      121          LD      A,(un_chr_atri)
308B 32 D4 45      122          LD      (chr_atri),A
308E 3A 0A 30      123          LD      A,(chr_OTHER)
3091 C9            124          RET
3092               125  ;
3092 1E 01         126  nxt_c5: LD      E,1
3094 3A 15 30      127          LD      A,(sp_chr_atri)
3097 32 D4 45      128          LD      (chr_atri),A
309A 3A 07 30      129          LD      A,(chr_TAB)
309D C9            130          RET
309E               131  ;
309E 1E 02         132  nxt_c6: LD      E,2
30A0 3A 15 30      133          LD      A,(sp_chr_atri)
30A3 32 D4 45      134          LD      (chr_atri),A
30A6 3A 08 30      135          LD      A,(chr_CR)
30A9 C9            136          RET
30AA               137  ;
30AA 21 0B 30      138  nxt_c7: LD      HL,str_EOF
30AD 3A 14 30      139          LD      A,(EOF_str_atri)
30B0 32 D4 45      140          LD      (chr_atri),A
30B3 C3 6E 30      141          JP      nxt_c4
30B6               142  ;
30B6 1E 02         143  nxt_c8: LD      E,2
30B8 3A 16 30      144          LD      A,(nm_chr_atri)
30BB 32 D4 45      145          LD      (chr_atri),A
30BE 3A 09 30      146          LD      A,(chr_SPACE)
30C1 C9            147          RET
30C2               148  ;
30C2 FE 01         149  nxt_c1: CP      1
30C4 CA CB 30      150          JP      Z,nxt_c2
30C7 3A 09 30      151          LD      A,(chr_SPACE)
30CA C9            152          RET
30CB               153  ;
30CB FD 7E 00      154  nxt_c2: LD      A,(IY)
30CE B7            155          OR      A
30CF C2 D8 30      156          JP      NZ,nxt_c3
30D2 FD 23         157          INC     IY
30D4 3A 09 30      158          LD      A,(chr_SPACE)
30D7 C9            159          RET
30D8               160  ;
30D8 1E 00         161  nxt_c3: LD      E,0
30DA 3A 16 30      162          LD      A,(nm_chr_atri)
30DD 32 D4 45      163          LD      (chr_atri),A
30E0 C3 6E 30      164          JP      nxt_c4
```

ントは，

1)　通常文字コードの場合は，バッファから文字コードを取り出すだけ

2)　タブコードはタブ位置にくるまでスペースを表示し続ける

3)　改行コードを表示したら，そのあとは画面右端までスペースを表示し続ける

4)　エンドコード(00_H)がきたらエンドコード文字列([EOF])にバッファポインタを移す

以上，4つのことです。

それでは，具体的に処理の流れを追いかけてみましょう。

まず，1)〜3)の処理の振り分けを行うために参照されるのが，Eレジスタです(振り分けは，105〜107行で行っています)。

・Eレジスタ=0の場合

まず，文字コードを取り出します(108行)。そしてバッファポインタとタブ情報テーブルポインタを進めます(109〜110行)。取り出した文字コードが，通常キャラクタかコントロールコードをチェックし，通常キャラクタならばサブルーチンからリターンします(111〜112行)。

次にある113〜120行では，タブ，改行，終端コードの判別とそれぞれの処理へジャンプしていて，それ以外のコントロールコードであればunknown characterとして処理されます(121〜124行)。

それぞれの処理では，タブ，改行コードであればEレジスタ(フラグ)の内容を書き換え，それぞれに応じたキャラクタをセットしてリターンしています(126〜136行)。終端コードであればバッファアドレスをstr_EOFに変え，アトリビュートを設定してから(138〜141行)，さらに文字列を取り出しに108行へジャンプしています。

このような構造だと終端コードで永久ループになりそうなものですが(なぜならたいていの文字列終端コードは00_Hであるからです)，そのへんはぬかりありません。EOFの文字列のエンドコードを$1A_H$と設定することにより，終了チェックを行っています(143〜147行)。ここで，なんでEレジスタに改行コードと同じフラグをセットしているのかがわかれば，完璧でしょう。

・Eレジスタ=1の場合

まず，107行から149行へジャンプして，さらに154行へジャンプします。154〜156行でタブコード位置まで表示が終わったかチェックし，終わっていなかったらタブ情報テーブルポインタを進めてから，スペースの文字コードを設定してリターン(157〜159行)。タブ位置に到達したら，Eレジスタを通常モードに戻し，さらにアトリビュートを通常文字コードに戻して，さらにバッファから文字コードを取り出すために108行へジャンプしています(161〜164行)。

・Eレジスタ=2の場合

Eレジスタ=1の場合と同じく，107行から149行へジャンプしています。そのあとは，Aレジスタにスペースの文字コードを設定してリターンしているだけです(151〜152行)。なぜここでEレジスタの初期化やアトリビュートの初期化を行わないか疑問に思うかもしれません。これは，ちょっと考えるとわかると思います。改行コードというものは，文字どおり次の行へ移ることを示す文字コードです。つまり，改行コードのあとには絶対に文字列を表示することがないのです。ま，当たり前ですね。

これで，FEのラインプリントルーチンの全貌がわかりました(うそつけー，わからないところもあったくせに，なんて突っ込みは無視)。最終的にどのようなルーチンを作ればいいのかまとめてみます。

1)　文字の色情報(アトリビュート)はchr_atriに格納されている

2)　表示すべき文字コードは，next_chrの返り値のAレジスタの内容を使う

3)　表示アドレスは，使用されていないBCレジスタに格納しておく

4)　表示アドレス計算は，画面の桁数に応じて変化させる

以上のことを守れば，FEのラインプリントルーチンを制作できます。よくわからなくて挑戦できなかった人も，これで大丈夫。安心して各機種専用のラインプリントルーチンを制作してくださいね。

バグフィクス

最後にFEで「ちょっとした不都合が見つかった」と制作者の松藤氏から報告がありました。修正のためのパッチ当てプログラムが届きましたのでここに紹介します(以下松藤氏の投稿原稿)。

コマンド入力において，画面の横幅を超えた入力を行うと画面が乱れるというバグが発見されました。動作上，それほど問題はありませんが，ちょっと見苦しいのでパッチ当てプログラムを送ります。

今回のプログラムでは，デバッグだけでなく，@シーケンスに対応することもできるようにしました(こっちのほうが嬉しい人が多いかな)。

まず，MACINTO-Cなどのツールを使ってリスト3のパッチ当てプログラムを打ち込んでください。チェックサムを確認後，

#S FE110.OBJ:8000:807E

以上のようにして，いったんデバイスにセーブしましょう。

次に，FE.OBJをメモリにロードしてください。あとは，

#J8000

のようにしてパッチ当てプログラムを起動するだけです。無事終了すると，

Complete.

の表示とともに終了します。パッチを当て終わったら，

#S FE.OBJ:3000:47FF:3000

以上のようにして，更新したFEをセーブし直しましょう。

さらに，@シーケンスを使用したい場合には，パッチ当てプログラム起動後，

3266　CD→C3

3267　C4→B6

以上の2バイトを変更してください。

なお，これまでに発表したプログラムのソースリストおよびオブジェクトプログラムはすべてフリーソフトウェアとします。活用してください。

リスト3

```
8000 11 4A 80 21 23 42 18 02 : 7B
8008 23 13 1A B7 28 0F BE 28 : 24
8010 F7 CD E2 1F 45 72 72 6F : 5D
8018 72 2E 0D 00 C9 11 4F 80 : 56
8020 1A B7 CA 2A 80 CD 39 80 : CB
8028 18 F6 CD E2 1F 43 6F 6D : FB
8030 70 6C 65 74 65 2E 0D 00 : 55
8038 C9 1A 47 13 1A 6F 13 1A : F3
8040 67 13 1A 77 13 23 05 20 : 66
8048 F9 C9 31 2E 30 30 00 13 : 94
8050 6D 45 CD 13 39 CD 0F 3B : E2
8058 3A CA 45 B7 C0 DD 36 01 : D4
8060 01 CD 9F 3E C9 03 60 32 : 09
8068 DC 6D 45 0C B6 32 CD C4 : 13
8070 32 FE 40 20 02 3E 1B C3 : AE
8078 69 32 01 25 42 31 00    : 34
----------------------------------
SUM: 87 E0 4E 88 76 22 F1 48 621E
```

リスト4

```
0000           1  ;=====================================
0000           2  ;
0000           3  ;     Screen Editor for S-OS SWORD
0000           4  ;     FE up to data v1.00 -> v1.10
0000           5  ;       Programed by H.Matsufuji
0000           6  ;
0000           7  ;=====================================
0000           8
0000           9
1FE2 P        10  _MPRNT  EQU     1FE2H
0000          11
3913 P        12  calc_view_offset        EQU     3913H
```

```
3B0F P              13  calc_locate_x          EQU     3B0FH
3E9F P              14  view_text              EQU     3E9FH
32C4 P              15  keyin                  EQU     32C4H
0000                16
41FB P              17  startup_message        EQU     41FBH
4223 P              18  version                EQU     4223H
0000                19
45CA P              20  flag_cmd_mode          EQU     45CAH
0000                21
0000                22
8000                23          ORG     8000H
8000                24
8000                25
8000                26  ;-----------------------------------
8000                27  ; Main
8000                28
8000                29
8000                30  main:
8000 11 4A 80       31          LD      DE,ver
8003 21 23 42       32          LD      HL,version
8006 18 02          33          JR      main9
8008 23             34  main8:  INC     HL
8009 13             35          INC     DE
800A 1A             36  main9:  LD      A,(DE)
800B B7             37          OR      A
800C 28 0F          38          JR      Z,main0
800E BE             39          CP      (HL)
800F 28 F7          40          JR      Z,main8
8011                41  ;
8011 CD E2 1F       42          CALL    _MPRNT
8014 45 72 72 6F    43          DB      "Error.",13,0
8018 72 2E 0D 00
801C C9             44          RET
801D                45
801D 11 4F 80       46  main0:  LD      DE,datas_of_patch
8020 1A             47  main1:  LD      A,(DE)
8021 B7             48          OR      A
8022 CA 2A 80       49          JP      Z,main2
8025 CD 39 80       50          CALL    patch
8028 18 F6          51          JR      main1
802A                52  ;
802A CD E2 1F       53  main2:  CALL    _MPRNT
802D 43 6F 6D 70    54          DB      "Complete.",13,0
8031 6C 65 74 65
8035 2E 0D 00
8038 C9             55          RET
8039                56
8039                57
8039                58  patch:
8039 1A             59          LD      A,(DE)
803A 47             60          LD      B,A
803B 13             61          INC     DE
803C 1A             62          LD      A,(DE)
803D 6F             63          LD      L,A
803E 13             64          INC     DE
803F 1A             65          LD      A,(DE)
8040 67             66          LD      H,A
8041 13             67          INC     DE
8042 1A             68  patch1: LD      A,(DE)
8043 77             69          LD      (HL),A
8044 13             70          INC     DE
8045 23             71          INC     HL
8046 05             72          DEC     B
8047 20 F9          73          JR      NZ,patch1
8049 C9             74          RET
804A                75
804A                76
804A 31 2E 30 30    77  ver:    DB      "1.00",0
804E 00
804F                78
804F                79
804F                80  ;-----------------------------------
804F                81  ; Patch datas
804F                82
804F                83
804F                84  datas_of_patch:
804F                85
804F                86
0013 P              87  patch1_length   EQU     patch1_end - patch1_start
804F                88  ;
804F 13             89          DB      patch1_length
8050 6D 45          90          DW      4580H - patch1_length
8052                91  ;
8052                92  patch1_start:
8052 CD 13 39       93          CALL    calc_view_offset
8055 CD 0F 3B       94          CALL    calc_locate_x
8058 3A CA 45       95          LD      A,(flag_cmd_mode)
805B B7             96          OR      A
805C C0             97          RET     NZ
805D DD 36 01 01    98          LD      (IX+1),1
8061 CD 9F 3E       99          CALL    view_text
8064 C9            100          RET
8065               101  patch1_end:
8065               102
8065               103
0003 P             104  patch2_length   EQU     patch2_end - patch2_start
8065               105  ;
8065 03            106          DB      patch2_length
8066 60 32         107          DW      3260H
8068               108  ;
8068               109  patch2_start:
8068 DC 6D 45      110          CALL    C,4580H - patch1_length
806B               111  patch2_end:
806B               112
806B               113
000C P             114  patch3_length   EQU     patch3_end - patch3_start
806B               115  ;
806B 0C            116          DB      patch3_length
806C B6 32         117          DW      32B6H
806E               118  ;
806E               119  patch3_start:
806E CD C4 32      120          CALL    keyin
8071 FE 40         121          CP      "@"
8073 20 02         122          JR      NZ,p31
8075 3E 1B         123          LD      A,1BH
8077 C3 69 32      124  p31:    JP      3269H
807A               125  patch3_end:
807A               126
807A               127
0001 P             128  patch4_length   EQU     patch4_end - patch4_start
807A               129  ;
807A 01            130          DB      patch4_length
807B 25 42         131          DW      startup_message + 42
807D               132  ;
807D               133  patch4_start:
807D 31            134          DB      "1"
807E               135  patch4_end:
807E               136
807E               137
807E 00            138          DB      0
807F               139
807F               140
807F               141  ; FE up to data v1.00 -> 1.10
807F               142  ; June 25,1995. by H.Matsufuji
```

THE SENTINELの続き

THE SENTINELがゲームにおけるキー操作の説明に独占されてしまったので，ここでそのほかのS-OS“SWORD”情報をお届けします。

●今月のフリーソフト

本文にもありますが，今月掲載した「MISSILE SYSTEM」，そして7月号掲載の「FE ver.1.0」がコピーフリーとなりました。制作者の吉田，松藤さん，ご協力感謝します。

THE SENTINELでは，引き続きS-OS“MOOK”化計画に伴う，フリーソフト化計画を実施しています。いままで掲載されたS-OS“MOOK”用アプリケーションで「著作権は放棄しないけど，オブジェクトをコピーして配布してもいいよ」という人がいらっしゃいましたら，アンケートハガキでご連絡ください。

よろしくお願いします。

●ラインプリントルーチン募集

今月の「FE ver.1.0ラインプリントルーチンの詳細」で，FEで使われていたラインプリントルーチンの中身が，だいたい明らかにされました。

エディタといえば，操作感覚が大切です。さすがにS-OS専用ルーチンでは，表示が重いのでFEを快適に使用することはできません。そのため，ぜひとも専用のラインプリントルーチンが必要です。

今月紹介された情報をもとに読者の皆さんが制作した，各機種専用のラインプリントルーチンの投稿を待っています。

投稿がありしだい，掲載していこうと思っています。

●S-OS“SWORD”MOOK化計画の話

ぼちぼち読者の皆さんに協力して打ち込んでもらったぶんのマニュアルが，整理できそうな気配です。あとは，アプリケーションを収集してディスクに放り込めば，オンラインマニュアルつきのS-OS“SWORD”MOOK（アプリケーション編）の試作版ができあがり！　となるでしょう。予定は未定ということで予断を許しませんが，これからも，作業の進行状況をお伝えしていきます。

●来月は？

そして，来月はどうやらSLANG用のパズルゲーム「CUBE」が登場する予定です。

ちょっとだけゲーム内容を紹介しましょう。画面上にいくつか種類のあるブロック（CUBE）を移動して，同じブロックで違うブロックを挟むと挟まれたブロックが消え，目的のブロックをうまく消せば，ステージクリアとなるものです。じっくり考える思考型のパズルゲームということで，頭を悩ませるのが好きな方は楽しみにしていてください。

どうやらX68000版も同時に制作されているようなので，S-OSユーザーでないX68000ユーザーも楽しみにしていてください。

●ピコッとな

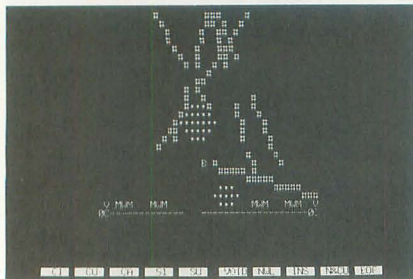
複雑怪奇になりがちな現在のゲームですが，やはりゲームの原点はピコピコにあります。基本を押さえたものであれば，シンプルなものでも面白いことは，読者の皆さんもご承知のとおりです。

ということで，最近ピコピコゲームをテーマにした企画をやりたい，そんな思いが頭の中をグルグルしています（なにをやるかは決まっていない。がんばってただの妄想に終わらないようにちゃんとした企画を立てなきゃ）。読者の皆さんのピコピコゲームに関する意見，要望，思い入れなどありましたら，アンケートハガキでお聞かせください。

全機種共通
S-OS"SWORD"要

MISSILE SYSTEM

Yoshida Masayuki
吉田 昌之

降り注ぐミサイルから都市を守れ！ フルキーを使う特異な操作にパニックしながら，美しい爆弾の花を咲かせるのだ。根性があるなら，ノーウェイトでバリバリ遊びましょう。

今回は，以前Oh!Xで「世紀末大戦術」という名で発表されていた，某有名ゲームのリメイク版といえるゲームです。

実際には，私の技量，時間，ゲームバランスなどを天秤にかけた結果，ゲームのルールはかなり変わってしまいました。しかし，やってみれば紛れもなく「あのゲームだ」とわかると思います。

このゲームは，トラックボールで遊ぶことを前提としたゲームなので「上下左右にカーソル移動して，という操作では，面白味が半減してしまう」と考えました。

「きちんと操作すれば，間に合わないはずないのに」というジレンマ的な部分が面白いと思うのです。

そこで，キーボードだけで，同様の感覚が味わえる，特殊な操作方法を採用しました。この方法を使えば，テンキーのない機種でも問題なく遊べますし，同時キー入力の問題もありません(どんな方法かはあとで説明します)。

入力方法

MACINTO-Cなどのダンプ入力ツールを使用し，リストを入力してください。チェックサムを確認後，

#S　MISSILE:A000:A87F:A000

として，いったんデバイスにセーブしてください。

そして，プログラムをメモリに読み込んだら，

#JA000

で実行できます。

ゲーム内容

上空から降り注ぐミサイルから，地上にある6つの都市を守ってください。

プレイヤーの攻撃も，敵と同じミサイルです。ミサイルの爆発で，敵ミサイルを巻き込み誘爆させることができます。

ミサイル発射台は，左右両端の2カ所にあり，それぞれ15発のミサイルが配置されています。一度に撃てるのは，それぞれ1発ずつです。

一定数のミサイルが降り終わると，1 ROUND終了で，ミサイルと都市の残数に応じて，ボーナス点が入ります。

この時点で，都市を規定数守り切れなかった場合，ゲームオーバーとなります。

敵ミサイルは，ROUNDが進むほど，速く，多くなっていきます。

操作方法

実行すると，B-GALET同様にシンプルなタイトル画面となります。スペースキーでゲームがスタートし，"Q"キーでOSに戻ります。そして，"+"キーでゲーム中のウェイト増加，"−"キーでウェイト減少となります（デフォルトは4)。

ゲームが始まると，画面下部に都市"MWM"が並び，両端にミサイル発射台が表示され，最低限守らなければならない都市の数が表示されます。

発射台の下の数字（16進数表示）は，残りのミサイル数を表しており，0になると発射できなくなってしまいます。

ゲーム中の使用キーは，メインキーのほぼ全部です(笑)。つまり，画面上の位置とキーボードの位置が，だいたい一致するようになっているのです。たとえば"1"キーを押せば，一番左上に向かってミサイルが飛びます。

テーブルサイズの都合により，大文字でしか受けつけませんので，CAPSキーはロックの状態でプレイしてください。

なお，ミサイル発射キーを押すとミサイルは，そのとき発射可能な発射台で近いほうを自動的に選びます。

攻略のコツ

とにかく，操作に慣れるまでそうとう辛いゲームです。

ブラインドタッチができるできない，なんてのはあまり関係ありません。とにかく慣れるしかありません。最初は，キーボード全体を使おうとせず，横一列分だけを使って狙うようにするとよいでしょう。

まず，QWERTYの段だけを使用し，極力この高さで撃ち落とすようにします。

以降，突破されるたびに，一段ずつ使用する列を下げていくようにすると，比較的狙いやすいと思います（まあ，実際にはそうそううまくいかないのですが)。

また，このゲームの難易度は青天井です

が，こちらのミサイル数は毎ROUND30発固定で増えることはありません。

物理的に限界がありますので，そこに到達するまでに，いかに効率よく点数が稼げるかがポイントのゲームです。

プログラムについて

ミサイルの移動ルーチンは，Oh!X 1994年7月号の「ゲーム作りのノウハウ・DDAアルゴリズム」を参考にしました。

このアルゴリズムの概念自体は，昔から知っていたのですが，割り算を使わずに計算できる方法があることを初めて知り，大変勉強になりました。

今回はこれを元に，移動とそのスピード調整，当たり判定と爆発をまとめて制御する汎用ルーチンを，休みのまとまった時間で一気に作成してしまい，とりあえず絵が動くようにしてしまいました。

目に見える部分が動くようになると，デバッグはもちろん，精神的にも非常に楽になります。あとは，また少しずつ追加していって完成させました。

しかし，表示にカーソルコードを混ぜていたり，当たり判定に#SCRNを使用したりと，処理速度において，OSの性能にかなり依存しています。

こういった使い方は，S-OSの作法としてはよいのかどうかはわかりませんが，これらの機能のおかげでずいぶんと楽をさせてもらっています。

終わりに

原作に比べ，敵の弾種は1種類のみだったり，連射がほとんどできないために，壁を作ることができなかったり，ミサイル発射台は壊れなかったりと，省略されたルールも目立ちますが，このゲームは操作系がすべてです。

この操作に慣れていき，極めていく過程を楽しむゲームですので，がんばって入力して，パニクッてください。

なお，今回の「MISSILE SYSTEM」もコピーフリー扱いとします。ひとりでも多くの人にこのゲームを遊んでもらえることを願っています。

表　線分移動ルーチン

IX+0	表示キャラクタコード， 00_H：未使用，FF_H移動終了
IX+1	現座標X
IX+2	現座標Y
IX+3	開始座標X
IX+4	開始座標Y
IX+5	終点座標X
IX+6	終点座標Y
IX+7	VX　Xは+か-か
IX+8	XY　Yは+か-か
IX+9	WXY　XとYどちらが移動基準 (常時+1)か，0=X，1=Y
IX+A	WX　X幅
IX+B	WY　Y幅
IX+C	e　計算作業用変数
IX+D	移動カウント
IX+E	移動スピード
IX+F	移動スピードカウント

リスト

```
A000 3E 28 CD 30 20 CD F1 A6 : E7
A008 21 00 00 22 22 A8 2E 10 : 4B
A010 22 20 A8 3E 0C CD F4 1F : 14
A018 21 0C 08 CD 1E 20 CD E2 : EF
A020 1F 4D 49 53 53 49 4C 45 : 35
A028 20 43 4F 4D 4D 41 4E 44 : 1F
A030 00 21 08 0A CD 1E 20 CD : 0B
A038 E2 1F 50 55 53 48 20 53 : B4
A040 50 41 43 45 20 4B 45 59 : 22
A048 20 54 4F 20 53 54 41 52 : 1D
A050 54 00 21 0F 0F CD 1E 20 : 9E
A058 CD E2 1F 53 43 4F 52 45 : 4A
A060 3A 00 2A 22 A8 CD BE 1F : D8
A068 21 0D 11 CD 1E 20 CD E2 : F9
A070 1F 48 49 53 43 4F 52 45 : 2C
A078 3A 00 2A 20 A8 CD BE 1F : D6
-----------------------------------
SUM: 08 F0 ED 85 A2 16 4B D5 6C6B

A080 CD D0 1F FE 51 C8 FE 2B : FC
A088 F5 CC A1 A0 F1 FE 2D F5 : 13
A090 CC C3 A0 F1 FE 20 20 E8 : 46
A098 CD D6 A3 CD CE A0 C3 13 : 57
A0A0 A0 3A 2F A8 3C FE 10 C8 : C3
A0A8 32 2F A8 F5 21 0A 14 CD : 0A
A0B0 1E 20 CD E2 1F 57 41 49 : ED
A0B8 54 20 00 F1 CD C1 1F CD : DF
A0C0 BA A6 C9 3A 2F A8 B7 C8 : B9
A0C8 3D 32 2F A8 18 DD CD C8 : D0
A0D0 A2 CD 35 A2 28 2A CD BA : 1F
A0D8 A5 CD EE A4 CD ED A0 CD : 2B
A0E0 D0 1F FE 71 C8 FE 70 CC : 60
A0E8 93 A6 C3 D1 A0 E5 F5 3A : 81
A0F0 2F A8 B7 28 08 67 2E 00 : 53
A0F8 2B 7D B4 20 FB F1 E1 C9 : 12
-----------------------------------
SUM: 9A 3A EE 7E FE 7D F7 AC 706E

A100 21 0C 08 CD 1E 20 CD E2 : EF
A108 1F 52 45 53 54 20 4D 49 : 13
A110 53 53 49 4C 45 3A 00 3A : F4
A118 31 A8 47 3A 30 A8 80 47 : F9
A120 CD C1 1F CD FF A1 21 0D : 48
A128 0A CD 1E 20 CD E2 1F 43 : 26
A130 49 54 59 20 42 4F 4E 55 : 4A
A138 53 3A 00 CD B2 A2 4F CD : CA
A140 C1 1F 41 3E 00 1E 0A 83 : 0A
A148 10 FD 47 CD FF A1 79 FE : 38
A150 06 20 28 21 07 0C CD 1E : 6D
A158 20 CD E2 1F 50 45 52 46 : 1B
A160 45 43 54 20 42 4F 4E 55 : 30
A168 53 21 20 31 30 30 20 70 : B5
A170 6F 69 6E 74 73 00 06 64 : 97
A178 CD FF A1 CD B2 A2 4F 3A : 17
-----------------------------------
SUM: 02 4A 88 5D 94 C7 DC 66 B323

A180 2E A8 91 CA 89 A1 D2 B5 : E2
A188 A1 21 0C 05 CD 1E 20 CD : AB
A190 E2 1F 2D 20 52 4F 55 4E : 92
A198 44 20 43 4C 45 41 52 20 : EB
A1A0 2D 00 3A 25 A8 3C 32 25 : C7
A1A8 A8 21 FF 4F 2B 7C B5 C2 : 35
A1B0 AC A1 C3 CE A0 3E 0C CD : 95
A1B8 F4 1F 21 0F 08 CD 1E 20 : 56
A1C0 CD E2 1F 47 41 4D 45 20 : 08
A1C8 20 4F 56 45 52 00 21 0F : 8C
A1D0 0F CD 1E 20 CD E2 1F 53 : 3B
A1D8 43 4F 52 45 3A 00 2A 22 : AF
A1E0 A8 CD BE 1F 21 FF 5F 2B : FC
A1E8 7C B5 C2 E7 A1 2A 22 A8 : 6F
A1F0 ED 5B 20 A8 B7 ED 52 D8 : DE
A1F8 2A 22 A8 22 20 A8 C9 D9 : 80
-----------------------------------
SUM: E4 35 57 4D 9B FF F5 EC 43C8

A200 21 0F 0F CD 1E 20 CD E2 : F9
A208 1F 53 43 4F 52 45 3A 00 : D5
A210 21 15 0F D9 2A 22 A8 CD : DF
A218 BE 1F 23 D9 CD 1E 20 D9 : BD
A220 CD BE 1F 10 F5 22 22 A8 : 9B
A228 CD BA A6 CD BA A6 CD BA : E1
A230 A6 CD BA A6 C9 3A 2B A8 : A9
A238 3D 28 04 32 2B A8 C9 3A : 71
A240 29 A8 3D 28 0C 32 29 A8 : 45
A248 3A 2A A8 32 2B A8 C3 74 : 48
A250 A2 CD A2 A2 C0 3A 2C A8 : 81
A258 B7 C8 CD B2 A2 B7 C8 3A : 59
A260 2C A8 3D 32 2C A8 3A 28 : 79
A268 A8 32 29 A8 3A 2A A8 32 : E9
A270 2B A8 B7 C9 CD D5 A6 6F : 0A
A278 26 00 16 16 CD C8 A6 47 : D4
-----------------------------------
SUM: 7D EC 8E EA A3 89 C0 DA 5001

A280 80 80 87 06 04 80 5F 3E : AE
A288 23 CD 1A A5 3A 2D A8 DD : 9B
A290 77 0E DD 77 0F DD 6E 05 : 38
A298 DD 66 06 CD 1E 20 CD F4 : 15
A2A0 1F C9 21 32 A8 11 10 00 : 04
A2A8 06 20 7E B7 C0 19 10 FA : 3E
A2B0 AF C9 21 04 17 01 00 06 : BB
A2B8 11 06 00 CD 1B 20 FE 57 : 74
A2C0 20 01 0C 19 10 F5 79 C9 : 8D
A2C8 CD FF A6 CD 77 A3 21 10 : 8A
A2D0 0A CD 1E 20 CD E2 1F 52 : 35
A2D8 4F 55 4E 44 20 00 3A 25 : B5
A2E0 A8 5F CD C1 1F 7B CB 3F : 39
A2E8 CB 3F 32 2C A8 4F 21 0B : 8B
A2F0 0C CD 1E 20 CD E2 1F 44 : 29
A2F8 45 46 45 4E 44 45 44 20 : 0B
-----------------------------------
SUM: E6 4C C4 4E 51 60 A2 69 EB56

A300 43 49 54 59 3A 00 79 3C : 28
A308 3C FE 06 38 02 3E 06 32 : F0
A310 2E A8 CD C1 1F 7B C6 06 : CA
A318 91 32 28 A8 32 29 A8 43 : D9
A320 3E 32 90 32 2A A8 32 2B : 61
A328 A8 7B 47 3E 1E 90 FA 35 : 85
A330 A3 20 02 3E 01 32 2D A8 : 0B
A338 3E 10 32 30 A8 32 31 A8 : 63
A340 21 FF 3F 2B 7C B5 C2 43 : C0
A348 A3 21 10 0A CD 1E 20 CD : B6
A350 E2 1F 20 20 20 20 20 20 : C1
A358 20 20 00 21 0B 0C CD 1E : 63
A360 20 CD E2 1F 20 20 20 20 : 6E
A368 20 20 20 20 20 20 20 20 : 00
A370 20 20 20 20 20 00 C9 3E : A7
A378 0C CD F4 1F 21 00 17 CD : F1
-----------------------------------
SUM: 37 37 DF CC 73 BD 66 00 8C5F

A380 1E 20 CD E2 1F 20 56 20 : A2
A388 4D 57 4D 20 20 20 4D 57 : F5
A390 4D 20 20 20 4D 57 4D 20 : BE
A398 20 20 4D 57 4D 20 20 20 : 91
A3A0 4D 57 4D 20 20 20 4D 57 : F5
A3A8 4D 20 20 56 20 30 46 2D : A6
A3B0 2D 2D 2D 2D 2D 2D 2D 2D : 68
A3B8 2D 2D 2D 2D 2D 2D 2D 2D : 68
A3C0 2D 2D 2D 2D 2D 2D 2D 2D : 68
A3C8 2D 2D 2D 2D 2D 2D 2D 2D : 68
A3D0 2D 2D 30 46 00 C9 21 00 : BA
A3D8 00 22 22 A8 3E 01 32 25 : 82
A3E0 A8 C9 DD 7E 00 D6 F0 DD : 6F
A3E8 34 00 87 6F 26 00 11 98 : F9
A3F0 A7 19 5E 23 56 DD 6E 01 : E3
A3F8 DD 66 02 CD 1E 20 CD E5 : 02
-----------------------------------
SUM: B3 79 BE 6E A5 58 E6 6F 6ED5

A400 1F C9 DD 6E 01 DD 66 02 : 79
A408 CD 1B 20 FE 2A CA 40 A4 : DE
A410 DD 35 0F C0 DD 7E 0E DD : 27
A418 77 0F CD 9B A4 DD 6E 01 : DE
```

```
A420 DD 66 02 CD 1B 20 FE 2A : 75
A428 CA 40 A4 FE 2D CA 4A A4 : 91
A430 CD 1E 20 DD 7E 00 CD F4 : 27
A438 1F DD 35 0D CA 4A A4 C9 : BF
A440 2A 22 A8 11 01 00 19 22 : 41
A448 22 A8 DD 7E 0D CD 63 A4 : 06
A450 DD 7E 02 FE 02 DA 5E A4 : 39
A458 3E F1 DD 77 00 C9 AF DD : D8
A460 77 00 C9 DD 6E 03 DD 66 : D1
A468 04 DD 5E 05 DD 56 06 DD : 5A
A470 75 01 DD 74 02 F5 3E 20 : 1C
A478 CD 31 A5 C1 DD 7E 0D 90 : 5C
---------------------------------
SUM: F7 11 E1 97 76 72 92 49 2F4F

A480 DD 77 0D CD 9B A4 DD 6E : B8
A488 01 DD 66 02 CD 1E 20 3E : 8F
A490 20 CD F4 1F DD 35 0D C2 : E1
A498 83 A4 C9 AF DD B6 09 C2 : FD
A4A0 C8 A4 DD 7E 01 DD 86 07 : 32
A4A8 DD 77 01 DD 7E 0C DD 96 : 2F
A4B0 0B F2 C4 A4 DD 86 0A DD : AF
A4B8 77 0C DD 7E 02 DD 86 08 : 4B
A4C0 DD 77 02 C9 DD 77 0C C9 : 48
A4C8 DD 7E 02 DD 86 08 DD 77 : 1C
A4D0 02 DD 7E 0C DD 96 0A F2 : D8
A4D8 EA A4 DD 86 0B DD 77 0C : 5C
A4E0 DD 7E 01 DD 86 07 DD 77 : 1A
A4E8 01 C9 DD 77 0C C9 06 20 : 19
A4F0 DD 21 32 A8 C5 DD 7E 00 : F8
A4F8 FE F0 CC 63 A4 DD 7E 00 : 1C
---------------------------------
SUM: 07 AC EA B1 C6 75 4F 87 48D9

A500 FE F1 D4 E2 A3 DD 7E 00 : A3
A508 FE EF D2 11 A5 B7 C4 02 : F2
A510 A4 11 10 00 DD 19 C1 10 : 8C
A518 DB C9 CD 9A A5 B7 C8 DD : 0C
A520 75 01 DD 74 02 DD 75 03 : 1E
A528 DD 74 04 DD 73 05 DD 72 : F9
A530 06 DD 77 00 7D BB D2 45 : A9
A538 A5 DD 36 07 01 7B 95 DD : AD
A540 77 0A C3 4D A5 DD 36 07 : 50
A548 FF 93 DD 77 0A 7C BA D2 : F8
A550 5E A5 DD 36 08 01 7A 94 : 2D
A558 DD 77 0B C3 66 A5 DD 36 : 40
A560 08 FF 92 DD 77 0B DD BE : 93
A568 0A D2 7E A5 DD 36 09 00 : 1B
A570 DD 7E 0A DD 77 0D CB 3F : D0
A578 DD 77 0C C3 8A A5 DD 36 : 65
---------------------------------
SUM: F5 68 BF C4 2F 6E 59 5C 7D43

A580 09 01 DD 77 0D CB 3F DD : 52
A588 77 0C DD 6E 01 DD 66 02 : 14
A590 CD 1E 20 DD 7E 00 CD F4 : 27
A598 1F C9 D5 C5 F5 DD 21 32 : A7
A5A0 A8 11 10 00 06 1E DD 7E : 48
A5A8 00 B7 CA B6 A5 DD 19 10 : E2
A5B0 F5 F1 C1 D1 AF C9 F1 C1 : A2
A5B8 D1 C9 CD D0 1F 4F 3A 24 : 03
A5C0 A8 B9 C8 79 32 24 A8 B7 : 57
A5C8 C8 FE 2C D8 FE 5F D0 F5 : EC
A5D0 D6 2C 87 26 00 6F 11 B8 : E7
A5D8 A7 19 5E 23 56 CD E1 A6 : EB
A5E0 CD C8 A6 3E 14 BB DA 0F : 31
A5E8 A6 DD 21 12 AA DD 7E 00 : BB
A5F0 B7 C2 FB A5 3A 31 A8 B7 : E3
A5F8 C2 35 A6 DD 21 22 AA DD : 44
---------------------------------
SUM: B3 0E 58 4A 99 42 C8 25 EDCA

A600 7E 00 B7 C2 0D A6 3A 30 : 14
A608 A8 B7 C2 63 A6 F1 C9 DD : C1
A610 21 22 AA DD 7E 00 B7 C2 : C1
A618 21 A6 3A 30 A8 B7 C2 63 : B5
A620 A6 DD 21 12 AA DD 7E 00 : BB
A628 B7 C2 33 A6 3A 31 A8 B7 : 1C
A630 C2 35 A6 F1 C9 3A 31 A8 : 6A
A638 3D 32 31 A8 21 00 18 CD : 4E
A640 1E 20 CD C1 1F 21 01 16 : 23
A648 3E 23 CD 1F A5 3E 03 DD : 10
A650 77 0E DD 77 0F DD 6E 05 : 38
A658 DD 66 06 CD 1E 20 F1 CD : 12
A660 F4 1F C9 3A 30 A8 3D 32 : 5D
A668 30 A8 21 25 18 CD 1E 20 : 41
A670 CD C1 1F 21 26 16 26 16 : 46
A678 3E 23 CD 1F A5 3E 03 DD : 10
---------------------------------
SUM: A3 E7 DB 46 AB BB D2 68 DD72

A680 77 0E DD 77 0F DD 6E 05 : 38
A688 DD 66 06 CD 1E 20 F1 CD : 12
A690 F4 1F C9 E5 21 11 18 CD : D8
A698 1E 20 CD E2 1F 50 41 55 : F2
A6A0 53 45 21 00 CD D0 1F FE : 73
A6A8 20 20 F9 CD 1E 20 CD E2 : F3
A6B0 1F 2D 2D 2D 2D 2D 2D 00 : 2D
A6B8 E1 C9 E5 F5 21 FF 0F 2B : DE
A6C0 7C B5 C2 BF A6 F1 E1 C9 : F3
A6C8 3A 27 A8 3C FE 06 20 01 : 6A
A6D0 AF 32 27 A8 C9 CD E1 A6 : CD
A6D8 3A 26 A8 E6 1F 06 04 80 : 97
A6E0 C9 08 C5 3A 26 A8 47 87 : 6C
A6E8 87 80 3C 32 26 A8 C1 08 : 0C
A6F0 C9 21 20 A8 11 21 A8 01 : 8D
A6F8 20 02 36 00 ED B0 C9 21 : DF
---------------------------------
SUM: B1 ED 35 97 7C 65 3F A0 E7D6

A700 32 A8 11 33 A8 01 00 02 : C9
A708 36 00 ED B0 C9 1E 2A 1F : 03
A710 1D 1D 2A 2A 2A 1F 1D 1D : 11
A718 2A 00 1E 1D 2A 2A 2A 1F : 02
A720 1D 1D 1D 1D 2A 2A 2A 2A : 1C
A728 2A 1F 1D 1D 1D 1D 2A 2A : 11
A730 2A 00 1E 1E 1D 2A 2A 2A : 01
A738 1F 1D 1D 1D 1D 2A 2A 2A : 11
A740 2A 2A 1F 1D 1D 1D 1D 1D : 04
A748 1D 2A 2A 2A 2A 2A 2A 2A : 43
A750 1F 1D 1D 1D 1D 1D 1D 2A : F7
A758 2A 2A 2A 2A 1F 1D 1D 1D : 1E
A760 1D 2A 2A 2A 00 1E 1E 1D : F4
A768 20 20 20 1F 1D 1D 1D 1D : F3
A770 20 20 20 20 20 1F 1D 1D : F9
A778 1D 1D 1D 1D 20 20 20 20 : F4
---------------------------------
SUM: 49 40 D2 B3 26 FE 12 0A 83B4

A780 20 20 20 1F 1D 1D 1D 1D : F3
A788 1D 1D 20 20 20 20 20 1F : F9
A790 1D 1D 1D 1D 20 20 20 00 : D4
A798 0D A7 0D A7 0D A7 1A A7 : DD
A7A0 1A A7 1A A7 1A A7 32 A7 : 1C
A7A8 32 A7 32 A7 32 A7 32 A7 : 64
A7B0 32 A7 32 A7 65 A7 65 A7 : CA
A7B8 1B 12 21 03 1E 12 21 12 : B4
A7C0 1E 03 03 03 06 03 09 03 : 3C
A7C8 0C 03 0F 03 12 03 15 03 : 4E
A7D0 18 03 1B 03 23 0D 20 0D : 96
A7D8 00 00 00 00 00 00 00 00 : 00
A7E0 22 08 05 0D 12 12 0C 12 : 7E
A7E8 0B 0D 0A 08 0E 0D 11 0D : 63
A7F0 14 0D 19 08 17 0D 1A 0D : 8D
A7F8 1D 0D 18 12 15 12 1C 08 : 9F
---------------------------------
SUM: A0 40 76 33 C0 5C F2 31 E998

A800 1F 08 04 08 0D 08 08 0D : 5D
A808 10 08 16 08 0F 12 07 08 : 66
A810 09 12 13 08 06 12 25 08 : 7B
A818 27 03 26 0D 24 03 00 12 : 96
A820 00 00 00 00 00 00 00 00 : 00
A828 00 00 00 00 00 00 00 04 : 04
A830 00 00 00 00 00 00 00 00 : 00
A838 00 00 00 00 00 00 00 00 : 00
A840 00 00 00 00 00 00 00 00 : 00
A848 00 00 00 00 00 00 00 00 : 00
A850 00 00 00 00 00 00 00 00 : 00
A858 00 00 00 00 00 00 00 00 : 00
A860 00 00 00 00 00 00 00 00 : 00
A868 00 00 00 00 00 00 00 00 : 00
A870 00 00 00 00 00 00 00 00 : 00
A878 00 00 00 00 00 00 00 00 : 00
---------------------------------
SUM: 5F 25 53 25 46 2F 34 33 92E7
```

▶ 全機種共通システムインデックス ◀

＊以下のアプリケーションは，基本システムであるS-OS“MACE”またはS-OS“SWORD”がないと動作しませんのでご注意ください。

▶2Mバイト増設しているX68000 CompactXVIユーザーとしては，その2Mバイトを捨てて8Mバイトにするのは，安いからといってできない。X68000 XVI/CompactXVIユーザーのためにもCZ-6BE2Bもオープン価格にして２万円くらいで買えるようにしてほしい。
大村　邦嘉(24)神奈川県

1987

■87年1月号
- 第35部　マシン語入力ツールMACINTO-C
- 連載　FuzzyBASIC料理法<4>

■87年2月号
- 第36部　アドベンチャーゲームMARMALADE
- 第37部　テキアペ作成ツールCONTEX

■87年3月号
- 第38部　魔法使いはアニメがお好き
- 第39部　アニメーションツールMAGE
- 付録　“SWORD”再掲載とMAGICの標準化

■87年4月号
- 第40部　INVADER GAME
- 第41部　TANGERINE

■87年5月号
- 第42部　S-OS“SWORD”変身セット
- 第43部　MZ-700用“SWORD”をQD対応に

■87年6月号
- インタラプト　コンパイラ物語
- 第44部　FuzzyBASICコンパイラ
- 第45部　エディタアセンブラZEDA-3

■87年7月号
- 第46部　STORY MASTER

■87年8月号
- 第47部　パズルゲーム碁石拾い
- 第48部　漢字出力パッケージJACKWRITE
- 特別付録　FM-7/77版S-OS“SWORD”

■87年9月号
- 第49部　リロケータブル逆アセンブラInside-R
- 特別付録　PC-8001/8801版S-OS“SWORD”

■87年10月号
- 第50部　tiny CORE WARS
- 第51部　FuzzyBASICコンパイラの拡張
- 第52部　X1turbo版S-OS“SWORD”

■87年11月号
- 序論　神話のなかのマイクロコンピュータ
- 付録　S-OSの仲間たち
- 第53部　もうひとつのFuzzyBASIC入門
- 第54部　ファイルアロケータ＆ローダ
- インタラプト　S-OSこちら集中治療室
- 第55部　BACK GAMMON

■87年12月号
- 第56部　タートルグラフィックパッケージTURTLE
- 第57部　X1turbo版“SWORD”アフターケア ラインプリントルーチン
- 特別付録　PASOPIA7版S-OS“SWORD”

1988

■88年1月号
- 第58部　FuzzyBASICコンパイラ・奥村版
- 付録　石上版コンパイラ拡張部の修正

■88年2月号
- 第59部　シューティングゲームELFES

■88年3月号
- 第60部　構造型コンパイラ言語SLANG

■88年4月号
- 第61部　デバッギングツールTRADE
- 第62部　シミュレーションウォーゲームWALRUS

■88年5月号
- 第63部　シューティングゲームELFES II
- 第64部　地底最大の作戦

■88年6月号
- 第65部　構造化言語SLANG入門(1)
- 第66部　Lisp-85用NAMPAシミュレーション

■88年7月号
- 第67部　マルチウィンドウドライバMW-1
- 連載　構造化言語SLANG入門(2)

■88年8月号
- 第68部　マルチウィンドウエディタWINER

■88年9月号
- 第69部　超小型エディタTED-750
- 第70部　アフターケアWINERの拡張

■88年10月号
- 第71部　SLANG用ファイル入出力ライブラリ
- 第72部　シューティングゲームMANKAI

■88年11月号
- 第73部　シューティングゲームELFES IV

■88年12月号
- 第74部　ソースジェネレータSOURCERY

1989

■89年1月号
- 第75部　パズルゲームLAST ONE
- 第76部　ブロックゲームFLICK

■89年2月号
- 第77部　高速エディタアセンブラREDA
- 特別付録　X1版S-OS“SWORD”<再掲載>

■89年3月号
- 第78部　Z80用浮動小数点演算パッケージSOROBAN

■89年4月号
- 第79部　SLANG用実数演算ライブラリ

■89年5月号
- 第80部　ソースジェネレータRING

■89年6月号
- 第81部　超小型コンパイラTTC

■89年7月号
- 第82部　TTC用パズルゲームTICBAN

■89年8月号
- 第83部　CP/M用ファイルコンバータ

■89年9月号
- 第84部　生物進化シミュレーションBUGS

■89年10月号
- 第85部　小型インタプリタ言語TTI

■89年11月号
- 第86部　TTI用パズルゲームPUSH BON!

■89年12月号
- 第87部　SLANG用リダイレクションライブラリDIO.LIB

1990

■90年1月号
- 第88部　SLANG用ゲームWORM KUN
- 特別付録　再掲載SLANGコンパイラ

■90年2月号
- 第89部　超小型コンパイラTTC++

■90年3月号
- 第90部　超多機能アセンブラOHM-Z80

■90年4月号
- 第91部　ファジィコンピュータシミュレーションI-MY

■90年5月号
- 第92部　インタプリタ言語STACK

■90年6月号
- 第93部　リロケータブルフォーマットの取り決め
- 第94部　STACK用ゲームSQUASH!
- 第95部　X68000対応S-OS“SWORD”
- 特別付録　PC-286対応S-OS“SWORD”

■90年7月号
- 第96部　リロケータブルアセンブラWZD

■90年8月号
- 第97部　リンカWLK

■90年9月号
- 第98部　BILLIARDS

■90年10月号
- 第99部　ライブラリアンWLB

■90年11月号
- 第100部　タブコード対応エディタEDC-T

■90年12月号
- 第101部　STACKコンパイラ

1991

■91年1月号
- 第102部　ブロックアクションゲームCOLUMNS

■91年2月号
- 第103部　ダイスゲームKISMET

■91年3月号
- 第104部　アクションゲームMUD BALLIN'

■91年4月号
- 第105部　SLANG用カードゲームDOBON

■91年5月号
- 第106部　実数型コンパイラ言語REAL

■91年6月号
- 第107部　Small-C処理系の移植

■91年7月号
- 第108部　REALソースリスト編

■91年8月号
- 第109部　Small-Cライブラリの移植

■91年9月号
- 第110部　SLANG用NEWファイル出力ライブラリ

■91年10月号
- 第111部　Small-C活用講座（初級編）

■91年11月号
- 第112部　Small-C活用講座（応用編）
- 第113部　MORTAL

■91年12月号
- 第114部　Small-C SLANGコンパチ関数

1992

■92年1月号
- 第115部　LINER

■92年2月号
- 第116部　シミュレーションゲームPOLANYI

■92年3月号
- 第117部　カードゲームKLONDIKE

■92年4月号
- 第118部　オプティマイザO80実践Small-C講座(1)

■92年5月号
- 第119部　COMMAND.OBJ実践Small-C講座(2)

■92年6月号
- 第120部　COMMAND.OBJ2実践Small-C講座(3)

■92年7月号
- 第121部　関数リファレンス実践Small-C講座(4)

■92年8月号
- 第122部　ワイルドカード実践Small-C講座(5)
- 第123部　グラフィックライブラリ GRAPH.LIB

■92年9月号
- 第124部　O-EDIT&MODCNV

■92年10月号
- 第125部　SLENDER HUL実践Small-C講座(6)

■92年11月号
- 第126部　EDIT実践Small-C講座(7)

■92年12月号
- 第127部　MAKE実践Small-C講座(8)

1993

■93年1月号
- 第128部　EDC-Tの拡張

■93年2月号
- 第129部　BLACK JACK

■93年3月号
- 第130部　シューティングゲームコアシステム作成法(1)

■93年4月号
- 第131部　シューティングゲームコアシステム作成法(2)

■93年5月号
- 第132部　シューティングゲームコアシステム作成法(3)

■93年6月号
- 第133部　REVERSI

■93年7月号
- 特別付録　MSX用S-OS“SWORD”

■93年8月号
- 第134部　MACINTO-C再掲載

■93年9月号
- 第135部　7並べ
- 特別付録　SLANG再々掲載

■93年10月号
- 第136部　シューティングゲームコアシステム作成法(4)

■93年11月号
- 第137部　S-OSで学ぶZ80マシン語講座(1)

■93年12月号
- 第138部　エディタアセンブラREDA再掲載

1994

■94年1月号
- 第139部　S-OSで学ぶZ80マシン語講座(2)

■94年2月号
- 第140部　YGCSver.0.20ユーザーマニュアル
- 第141部　S-OSで学ぶZ80マシン語講座(3)

■94年3月号
- 第142部　S-OSで学ぶZ80マシン語講座(4)

■94年4月号
- 第143部　S-OSで学ぶZ80マシン語講座(5)

■94年5月号
- 第144部　S-OSで学ぶZ80マシン語講座(6)

■94年6月号
- 第145部　YGCSver.0.30

■94年7月号
- 第146部　シューティングゲーム作成講座(1)

■94年8月号
- 第147部　シューティングゲーム作成講座(2)

■94年9月号
- 第148部　怪しいZ80の使い方(テクニック編)

■94年10月号
- 第149部　シューティングゲーム作成講座(3)
- 第150部　怪しいZ80の使い方(未定義命令編)

■94年11月号
- 第151部　B-GALET2

■94年12月号
- 第152部　シューティングゲーム作成講座(4)

▶いまだに「XL/Imageお試し版」で遊んでいるのは私くらいしかいないでしょうね。SLASHのPLGファイル→PPDのコンバータまで作ってフォンシェーディングをかけて喜んだりしています。
河合　章紀(24)茨城県

多様性と理性にまつわるミステリー

赤面写真に赤面

大学の出身学科の同窓会から立派な同窓会誌なるものが送られてきました。それには，この50年間に送り出された卒業生の顔写真が，近況などの文章とともにずらりと載っています。厚さ1cmにも及ぶ立派なものです。

それを手にした僕は，赤面してしまいました。なぜなら，自分の写真があまりにも目をひくものだったからです。目立ちたがりというよりは，勘違いが原因です。

幹事の人から写真入りの名簿を作るのでよろしくとの手紙が来たとき，これは同期の仲間内だけでパンフレットのようなものを手作りするのだと勝手に思い込んでしまいました。そして，手元に適当な写真がなかったので，パソコンの中にあった自分の顔写真データに背景をレタッチソフトでぼかすなどの加工をしてから，エプソンのお手軽なカラープリンタで紙に印刷したものを送ったのでした。アップルのお手軽なデジタルカメラで撮ったもので，酒を飲んで赤面している（ポーズはロダンの「考える人」）ものでした。

先日も夜寝る前にその同窓会誌を読んでいたら，あっという間に2時間ほどたってしまいました。そのなかで，かなりお歳を召されていて，しかも業種的にそれほど関係なさそうな人の文章にも「マルチメディア」とか「インターネット」という言葉がよく出てくるのはご時勢を感じさせます。

いやー，それにしても，さまざまな人生があるものです。だから読むのをやめられません。文章を読んで写真を見て「ふーん，こんな人生かー」などと勝手に妄想してしまいます。だいいち自分の同期生の30人弱を見たって，あまりに多様です。

私がいたところは，工学部の中の計測関係を専門とするコースでして，センサ，電子回路，制御，信号処理がメインです。計算機関連の私はちょっと邪道ともいえますが，それどころではありません。銀行マン，医者，宇宙（開発事業団）などは，むしろ普通です。なんと，カトリック司祭になろうとしている人とか，はたまた自衛隊で匍匐前進している人までいるのですから。

さらに，学生時代のアルバムを横にもってきて，この同窓会誌の写真と比較してみると，エステの広告ではありませんが，ついつい大笑いしちゃいます。同じ専門の授業を同じように受けていた連中がかくも多様な人生を送り，そしてさまざまな顔に変形していくのですね。

これが人生なのだ。ジーンっ（わしゃ，じじいだ）。

生態の多様性

いろんな人間がいることは，同窓会誌を何時間も楽しめる以外に，人間という種の存続を考えても貴重なことです。環境はきわめてダイナミックに変化していくものですから，多様な人間がいて初めてその変化に追随していけるというものです。

1995年8月20日の世田谷区で生きていくのに最適な人間がいたとしても，少し西のほうに移動したり，あるいは，少し時間が経てば，もう最適ではなくなるかもしれません。しかも，そのような環境の変化を我々は完全に予測することはできないのです。ですから，あまりにも狭い固定された観点から見ないで，多様性を積極的に評価していくことには大きな意味があります。

人間という種だけでなく生物種全体を考えてみましょう。それこそ，本当にさまざまな種が絶妙なバランスをとりながら，地球上に存在しています。さまざまな種がそれぞれの持ち場で棲息しながら，全体としてひとつの解となっているのでしょう。そのようなマクロな視点で，人間だけでなくほかの種も含めて考えると，あるいはほかの種から見ると，我々人間の個性による差などむしろほとんど無視できるほど小さなものなのかもしれません。

つい先日もテレビで言語学者の巨人チョムスキーが，「我々はすぐに小さな差ばかりに目がいくが，距離をおいて見るとその共通性の大きさがわかる。我々がカエルを見てさまざまな個性を認めないように，ほかの種が我々を見ればほとんど似たようなものだ」というようなことをいっていました。彼はどの言語にも共通するような，我々人間が先天的にもっている普遍文法の存在を主張しているのですが，述べられていることはもっともです。

多様性を数値で表す

生物種における多様性というものに生物学者は長い間注目してきました。そして，それを定量的に計る方法を考えてきました（文献1）。また，人工生命においても，多様性の起源や多様性の変動はその中心的なテーマです。

とくに文献2は人工生命の本質的な部分を捉えようとしている論文です。これは人工生命における仮説である「単純な計算機モデルは複雑な適応システムの本質的な特性を捉えることができる」という点に関して，単純な人工生命モデルのシミュレーション結果を用いて説明しています。

そのモデルでは平面上に自律的に振る舞う主体（エージェント）を多数棲息させます。そして，食料をうまく見つけることによって，適当な方向に適当なだけ移動するように進化するというものです。

各エージェントは自分の周りの各位置の餌の分布密度を感じるセンサをもっていて，それを基にどのような方向にどれだけの距離だけ進むかということを決めます。この戦略が遺伝子にコーディングされており，ダーウィン的な自然淘汰によって進化します。このモデルの説明自体はこの論文の主旨ではなく，このモデルを用いていくつかの尺度からこのモデルによる結果を捉え直すということに主眼をおいています。

その重要な尺度が「多様性」です。ここでは，各エージェントにおけるセンサ入力に対する出力の分散の度合いを多様性と定義しています。多様性は進化するにつれてしだいに増していくのですが，特に興味深いのはその増え方です。単調に増えていくのではなく，横ばいが続いてから，急に断

続的に増え，また横ばいが続いてからまた断続的に増えるということを繰り返すのです。

この原因についていろいろ分析していますが，グールドらの断続平衡進化説(なだらかに進化が進むのではなく，ときどきググっと急激な進化が繰り返されるというもの)と関連させて議論しています。確かに，一様に進化が進むのではなく，ときどき大進化が起こるということを間接的に裏づけていると解釈することもできるのでしょう。

反多様性の快感

多様性の重要性を我々はそれなりに理解しているわけですが，その一方で，なるべく画一的で整然とした状態に，大きな快感を覚えるのも事実です。

むろん，人が部屋をきれいに片づけるとなにやら嬉しいというのは，単に部屋を使うのに便利だという効率上の問題として考えることもできましょう。しかし，たとえば，テレビゲームなど純粋に快感を求めるものを考えてみるとクリアになります。

テトリスタイプのゲームはまさにきっちりと乱雑さをなくしていくことの快感を追究したゲームであるといえます。我々をテトリスに熱中させてしまうもの，それは生態における多様性とかエントロピー増大という物理法則に逆らうことの快感と関連があると思われてしまうのです。

我々が学問をしたり本を読んだりするのも，結局はこのような心理的特性と一致するのでしょう。なぜならば，テトリスや部屋の掃除などは，画面上だとか目の前にある物体を実際に移動させて整然とするということですが，この作業を頭の中でやるのが学問的な作業の目指す主要な目的といえるからです。いろいろな情報がばらばらと頭の中に入っているのではなく，整然と説明され，関連づけられているほうが快感であるというわけです。

チョムスキーがいうような，人はつい差ばかりに目がいってしまうということ，あるいは，閉鎖された社会だとややもすると集団が画一化されていき，しだいにエネルギーを失って衰退していくという傾向も，このような心理的な特性が働いてしまうのでしょう。残念なことではありますが。

妊娠小説

さまざまな感情の機微を微妙に描くことのできる芸術である小説，これをスパスパと大胆に片づけてくれる思いっきり気持ちのよい本(文献３)があります。

小説といっても「妊娠小説」に属する小説を対象にしているのですが，「妊娠」を「標準装備」した本をすべて含むと定義しているので，よく知られている古典的な名作もかなり含まれます。

たとえば，森鷗外の『舞姫』と島崎藤村の『新生』という一見まるで違う２つの小説は，妊娠小説として見るとまったく同じであるというのです。つまり，両者とも，

・地位も名誉も教養もある年長の男が
・なにももたないうんと年下の女を
・妊娠させて病にいたらしめて捨てて，
・最後に安泰を得た

という話なのだそうですから。

かなりの作業量を要したと思われる分析はきわめて心地よいものです。受胎告知の様式，妊娠効果の基礎知識などはもう見事としかいいようのない面白さです。クライマックスは妊娠物語の類型学というところで，８種類の小説に分類してしまいます。少しでも興味の湧いた人はぜひ読んでみてください。面白さは保証します。

ここまで理路整然と説明してくれると，自分も小説のひとつぐらい軽く書けそうな気さえしてきます。それどころか，計算機で小説を自動生成できるのではないかという無謀な思いつきまでも誘い出してくれるのですから。

「理性」と「真理」

納得のいく説明をほしがる，雑然ではなく整然を求める，散らかった部屋を片づけたくなる，テトリスのとがったパターンをはめこんで平らにして一段消して喜ぶ，このような心理的特性の根本にあるのは，やはり人間こそがもつと思われている「理性」のなせるわざなのでしょうか？

自然界における多様性は，むろん単なる雑然ではありません。また，生命というものも同様に，完全にランダムな世界の現象ではなく，完全な静寂との間の微妙な領域に生じている現象です。そのような究極の領域までをも単純なモデルで説明してわかった気になりたいという欲求の下には，理性という名前で表されるようななにかがあるのでしょう，たぶん。

しかし，これは重要なことであると思いますが，それはなんでもいいから「真理」がわかればいいというのとは違うのです。人間の頭で理解できるような真理を求めるのが理性であるのでしょう。

哲学者アランが1926年に書いた「理性」とタイトルのつけられた文章を僕はまたしても思い出してしまうのでした。最近のある教団にまつわる不幸を見せつけられるごとに思い出したのと同じように。

「……これらの理由により，……銀のスプーンやインクの沼を用いて重要な大真理を見せてやると約束されたときにも，そのようなものを見つめはしないであろう。また，回教僧のようにぐるぐるまわればなにか偉大な秘訣が得られるというときにも，そのようなことはしないであろう。言葉を変えれば，理性のほうが真理よりも重要だと私は信ずるのである」(文献４)

文献

１)A．E．Magurran，“Ecological Diversity and Its Measurement”，Princeton Paperbacks

２)M．A．Bedau，“Three Illustrations of Artificial Life's Working Hypothesis”，Evolution and Biocomputation（Lecture Notes in Computer Science 899)，53-68pp．

３)斎藤美奈子，『妊娠小説』，筑摩書房

４)アラン著，原亮吉訳，『人間論』，白水社

e-mailアドレス

ari@info.human.nagoya-u.ac.jp

NIFTY-ServeやPC VANから送信するときは，前者が，INET:，後者がINET#を上記のアドレスの前につける。

イワシとナスビの謎

Takazawa Kyoko
高沢 恭子

たまには旅に出てみると，意外なことに気がついたりします。たとえば，昔から耳にするフレーズが，なにを意味するかは知っていても，その由来についてまでは知らないことがほとんどですが……。

「一富士，二鷹，三ナスビ，っていうでしょう，いままであの意味がどうしてもわからなかったんですけどね」

夫の向かいの席から，H氏が明るく語りかけた。心身，頭脳，すべてが健康といった感じの人だ。

初夏の夕暮れ，鳥取市のある料亭でのこと。雨で美しさを増した庭園をのぞむ東南の角の和室で，私たち夫婦はH氏ご夫妻とお会いしていた。

「あれは，日本三大仇討ちに由来していたんですねぇ」

「ほうー」と，夫はいかにも感心した顔つきで相づちをうっている。

だまって聞きなさい

あまりに古い「一富士，二鷹……」のことわざ，あるいは言い伝え。初夢で見ると縁起がよいベストスリーと聞かされてきたけれど，無意味なコジツケくらいに思っていた。

それがいま「仇討ち」と聞いて，にわかにスジが通った気がした。

「それじゃ，タカは鷹の羽で忠臣蔵でしょうか」

私は思わずそういって，右にいる夫のほうを見た。夫はすました顔をして，H氏の話をさらにうながす姿勢でいる。

しまった，またヤッタと私は思った。たとえ自分がどんなによく知っている話であっても，心から耳をかたむけるのが，いつもの彼の作法だった。

それと反対に，ほんの少ししか知らないことですぐに口をはさむのは私だ。仇討ちで「タカ」なら，浅野氏の家紋「鷹の羽」くらい誰でも思いつくだろうに。

「そうです，四十七士の討ち入りです。それで，あとの2つですけどね」

H氏は夫と大学が同窓で，電子部品メーカー工場長として，1年前に鳥取に赴任されていた。

夫がしごとを通じてH氏とお会いするうちに，夫人もまた，私と大学が同窓であることがわかった。しかも，彼女が埼玉の家と鳥取とを往復している状況まで，私たちとよく似ている。

そんな奇遇から，今回の私設同窓会が私たち夫婦の旅行をかねて開催された。目の前には，お料理というより食べられる工芸品といったおもむきの，高価で珍しい調理の品々がつぎつぎに運ばれていた。

H氏は鳥取を中心に，近辺の土地の歴史や文化をたいへんよく研究されているらしい。それはどうも，新しい土地で責任者としてのポストにつく人たちにとって，重要なしごとのひとつであるようだ。

休日にはご夫妻で戸外に出かけて絵を描かれるそうで，任務地で楽しくすごされるようすにも共感をおぼえた。

JR又右衛門コース

「富士は曾我兄弟なんですよ」

「あー，そうかぁ」と思ったが，こんどはガマンして黙っていた。

曾我十郎(兄)と五郎(弟)は，富士の裾野で父の仇，工藤祐経を討ちとった。22歳と20歳だった。1194(建久4)年のことで，征夷大将軍源頼朝も兄弟の心意気に感服したという。

伊豆伊東の領地を従兄(伊東祐親)に奪われた工藤祐経は，その子祐泰を殺した。祐泰の2人の息子は当時5歳と3歳，再婚した母の姓，曾我を名乗った。仇討ちで兄は死に，残った弟の五郎を頼朝は赦免しようとした。しかし祐経の子の哀願により処刑した。兄弟は仇討ちを生きがいとした人生だったことになる。

「ところでナスビですよ」

さぁて，ナスビとは野菜のナスであることはたしかなようだが，仇討ちでナスとはなんだろう。

3つのなかで，いちばん怪しく，ぜひとも究明したい意欲をおこさせるのは，このナスビにちがいない。

ナスビ，なすび，なんだろう。

「荒木又右衛門だったんですねぇ」

アラキ・マタエモン。宮本武蔵は知っていても「剣豪荒木又右衛門」の名を知る若い人は，もうとても少ないそうだ。そのアラキとナスビの関連はなにか。

「鍵屋の辻の決闘という，あれですよ。たいへんな仇討ちのあげく，事を成したという意味から，なす，ナスビになったんだそうです」そして，「又右衛門は伊賀上野で生まれて，鳥取で死んでいるんですねぇ」とH氏はいった。

三重県の伊賀上野城下「鍵屋の辻」で義弟を助けて仇討ちし，のちに鳥取藩に身柄を預けられて半月後に急死したという。

伊賀上野のマンションをあとにして，鳥取をおとずれた私たちは，知らずに又右衛門コースを体験したことになる。

又右衛門のお墓はこの近くの玄忠寺にあり，ナスビの由来も寺の住職から聞いた話だそうだ。

「しかし，又右衛門の36人斬りというのはウソのようですね。そんなには斬れないらしいです」H氏がつけくわえた。

「そーですよね，ホームランを続けて36本打つようなもんですものね」

またよけいなことをいったかなと反省していると，私の向かいにすわられていた夫人が，お料理についていろいろと説明をし

てくださった。季節によって材料もさまざまに，意趣をこらした献立を楽しませてくれるのだという。

昼間おとずれたとき砂丘を濡らしていた同じ雨が，あいかわらず，静かに面前の庭の緑を潤している。

オトナ100円の資料館

「鍵屋の辻」と呼ばれる史跡は，三重のマンションからクルマで5分もかからないところにある。

伊勢と奈良への分岐点を示す石の道標が往時のままで，決闘の現場は，街道ぞいの木立にかこまれた静かな場所である。そこで，のちに「三大仇討ちのひとつ」とされるような激しい戦いがあったとは。

ここには資料館もあって，じつはいままで2回も見学したことがあるのだが，事件の概要がさっぱりつかめなかった。

だがナスビのルーツとされるほどの大事件なら，全容を知らないままにはしたくない。ふたたび出かけていき，資料を熟読してみた。

3度目の挑戦でいろいろとわかった。

1630（寛永7）年に備前岡山の池田藩で，家臣の河合又五郎が同輩の渡辺源太夫を口論のあげく殺害した。19歳と17歳だった。

後世に残る仇討ちの原因は，意外にも若い男性同士の同性愛と嫉妬による言い争いだった。

又五郎は逃亡し，藩主の池田忠雄公が捜索を命じると，江戸の旗本安藤家にかくまわれていることがわかった。

安藤家は又五郎の父が家老をつとめていたが，その父は罪をおかして主君から斬首される寸前に池田藩の大名行列のなかに逃げ込み，保護されていた。

池田藩は安藤家に又五郎の引きわたしをもとめたところ，安藤家からは又五郎の父を差し出すよう要求があった。そこで，両者を交換する協定ができたのだが，池田藩が又五郎の父を差し出したものの，安藤家は又五郎を引きわたさなかった。

これが大名と旗本の対決というかたちで社会的な事件ともなり，当時の人々の関心の的になった。

折悪しく池田公が急逝。騒動の罪から池田藩は幼君とともに鳥取に国替えを命じられる。無念の池田藩。又五郎に弟を殺された渡辺数馬は，又五郎の上意討ちを願い出て許可される。

この仇討ちのために，渡辺数馬が助太刀をもとめたのが荒木又右衛門である。

おツリがたりない話

荒木又右衛門は伊賀に生まれ，はじめ池田藩に仕えたが，のちに柳生十兵衛に師事して柳生新蔭流を極める。その後，大和郡山藩に仕えていた。

妻の弟である渡辺数馬から助力をもとめられるが，いったんはことわる。人に依存するような仇討ちは成功できないと思ったからだそうだが，義弟の熱意を汲んで承諾する。仇討ちにあたり，大和郡山藩指南役の職をしりぞく。

河合又五郎は，旗本側の武芸者たちに警護されながら諸国を逃げまわっていたが，江戸に下るルートで伊賀を通過する情報をキャッチされる。

荒木又右衛門と数馬の一行4名は，伊賀上野の城下，鍵屋の辻にある茶屋で待ち伏せた。事件発端から4年後の寛永11年11月7日の早朝だった。

又五郎の一行は武芸の達人をふくむ総勢11人，又右衛門は敵の援軍の主力をおおかた殺傷したあと，数馬と又五郎の一騎討ちを見守った。数時間にもおよぶ激闘の末，数馬は本懐をとげる。

数馬と又右衛門の武勇はたたえられ，決闘は国じゅうの話題となった。

鍵屋の辻には，寛永の当時から営業を続けている茶屋があり，店内の柱には，斬り合いの刀キズが深々と残されている。

又右衛門一行が仇討ちの前に食べたのはそばとイワシの目刺しだった。この食事の代金の支払いを済ませて店を出た又右衛門が，1文だけ不足していたおツリを取りにもどった話は，この仇討ちの名高いエピソードとなっている。

たった1文ではあるが，仇討ちで平静を失い気づかなかったと思われては武士の恥と考えた，という解釈があるようだ。

後世の人たちはなにかと脚色したり，意味づけをしてストーリーをふくらませるのが好きだ。

仇討ち前の食事が，なぜイワシなのかという説明が，茶屋の壁にあった。

イワシは，伊賀の方言「ゆわす」が事を成就させる意味であることから，それになぞらえたのだということだ。ナスビの成すと同じだったのだ。

三哲，三筆，三聖人。三役，御三家，三奉行。三強，三大，三美人。

3つにまつわるものは，どこにでもあるらしい。3はムダがなくてまとまりがよい。

危険なドライバーを，「一姫（女性），二虎（飲酒），三ダンプ」といったり，大衆の好みがこどものようだというので，「巨人，タイホウ，たまごやき」というのもあった。いまなら，「巨人，ワカタカ，ハンバーガー」だろうか。

3つの言葉はほんとうの順位よりも，リズムで配置されることも多い。3つ目の言葉がいちばん長くなる。

だから，その3つがなにかは問題ではなく，「一富士（通），二マック，三シャープ」なんてことにきっとなる。

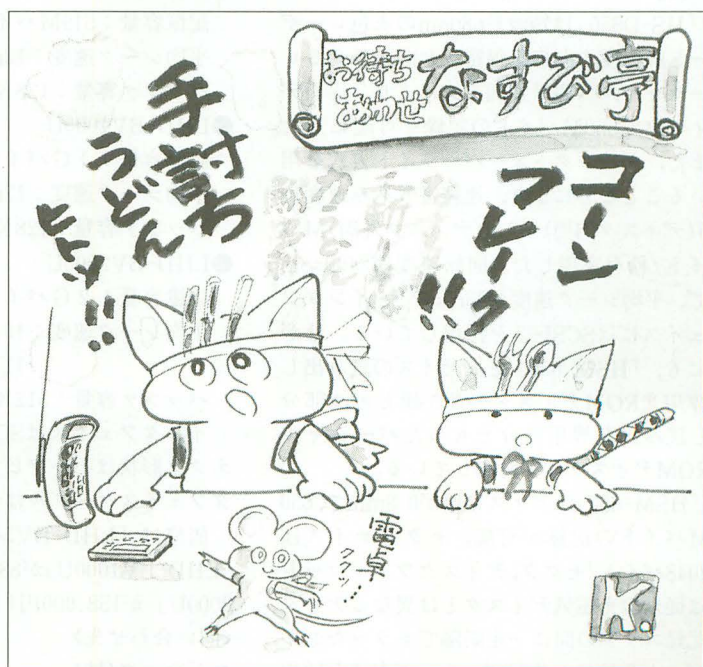



illustration : Kyoko Takazawa

PENGUIN INFORMATION CORNER

ペ・ン・ギ・ン・情・報・コ・ー・ナ・ー

NEW PRODUCTS

HSドライブ/HSディスク
HS-D650/HSM-650
ソニー

HS-D650

HSM-650

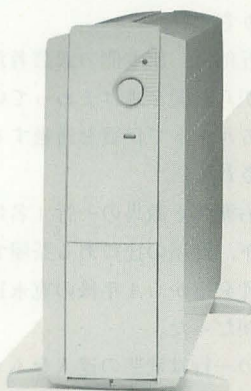
ソニーは3.5インチ光磁気ディスクHSドライブ「HS-D650」，HSディスク「HSM-650」のサンプル出荷を始めた。

「HS-D650」は短波長680nmの赤色レーザーと線密度方向の記録密度を向上させるレーザーパルス磁界変調方式により，片面ディスクで650Mバイトの記録を可能にした。また，ダイレクトオーバーライト方式を用いることなどにより，連続書き込み速度1.0(ディスク内周)～2.0(ディスク外周)Mバイト/秒を実現した。回転速度は2400rpmで，平均シーク速度が33ms以下，インタフェイスにはSCSI-2を採用している。ほかにも，「HSM-650」と同サイズの読み出し専用光ROMディスクや書き換え可能部分と読み出し専用部分をもったパーシャルROMディスクにも対応している。

「HSM-650」はディスク厚が0.8mmで，650Mバイトの記録が可能。セクタサイズは2048バイト/セクタ。ディスクフォーマットは従来の光磁気ディスクとは異なるデータ記録エリアの間に一定間隔でトラッキングピットを設けるサンプルサーボ方式を採用している。

価格は「HS-D650」が200,000円，「HSM-650」が17,000円(ともにサンプル価格)。

〈問い合わせ先〉
ソニー(株)　☎03(5448)3311

ハードディスクドライブ
LHD-BV540U/1000U/2000U
ロジテック

LHD-BV2000U

ロジテックはハードディスクドライブ「LHD-BV540U」「LHD-BV1000U」「LHD-BV2000U」の3機種を発売した。

●LHD-BV540
記憶容量：519Mバイト
平均シーク速度：12ms
バッファ容量：128Kバイト

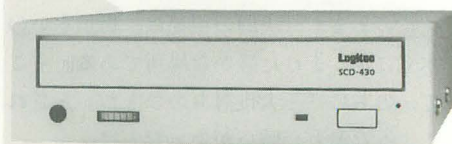
●LHD-BV1000U
記憶容量：1Gバイト
平均シーク速度：12ms
バッファ容量：128Kバイト

●LHD-BV2000U
記憶容量：2Gバイト
平均シーク速度：10.4ms(リード)
11.4ms(ライト)
バッファ容量：512Kバイト

インタフェイスはSCSI-2を採用し，コネクタ形状はハーフピッチ50ピンで，インタフェイスケーブルは付属している。

価格は「LHD-BV540U」が55,000円，「LHD-BV1000U」が88,000円，「LHD-BV2000U」が158,000円(ともに税別)。

〈問い合わせ先〉
ロジテック(株)　☎03(3251)3271

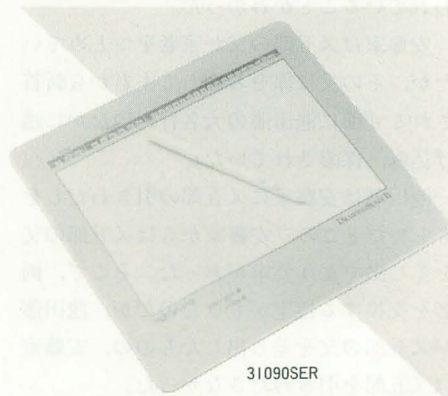
CD-ROMドライブ
SCD-430
ロジテック

SCD-430

ロジテックは4倍速CD-ROMドライブ「SCD-430」を発売した。

本機の転送速度は4倍速時で600Kバイト/秒で，バッファ容量は256Kバイトを搭載している。ディスクセット方式はトレイ方式，インタフェイスはSCSI-2を採用し，コネクタ形状はハーフピッチ50ピン。規格はCD-ROM XA，PHOTO CD，CD-DAに対応している。

価格は29,800円(税別)。

〈問い合わせ先〉
ロジテック(株)　☎03(3251)3271

タブレット“DRAWINGSLATEII”
31090SER/32120SER
エヌエス・カルコンプ

31090SER

エヌエス・カルコンプはタブレット“DRAWINGSLATEII”「31090SER」「32120SER」2機種を発売した。

「31090SER」は描画領域が152.4mm(縦)×228.6mm(横)で，ペンの筆圧だけでなく傾きや高さを認識するAFT機能を搭載して

いる。分解能は2540LPIで，読み取り方式は電磁誘導方式を採用。X68000との接続はRS-232Cを使用。大きさは247mm(縦)×279mm(横)×4.5mm(厚さ)で，重さが600g。

「32120SER」は描画領域が304.8mm(縦)×304.8mm(横)で，大きさが416.6mm(縦)×447mm(横)×4.5mm(厚さ)，重さが1040g。そのほかの機能は「31090SER」と同じ。

両機の付属品は筆圧対応のスタイラスペン，ACアダプタ，各機種用のドライバとアプリケーションなど。X68000で対応しているソフトは「MATIER ver.2.1」「Hiper Pixel Works ver.2.0」がある。

価格は「31090SER」が34,800円，「32120SER」が79,800円(ともに税別)。

〈問い合わせ先〉
エヌエス・カルコンプ(株)　☎03(3355)8911

レーザープリンタ LBP-730 キヤノン

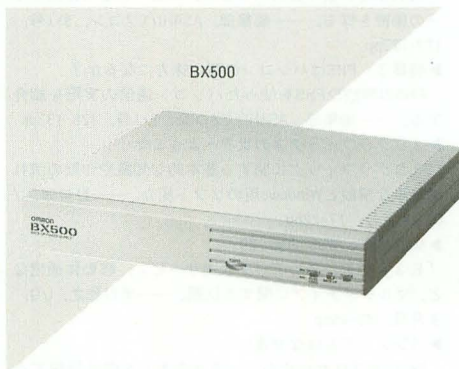
LBP-730

キヤノンはレーザープリンタ「LBP-730」を発売した。

本機は省エネルギーとウォームアップレスを実現するオンデマンド定着方式を採用することで，出力までの時間を短縮した(A4用紙をカセット給紙した場合，約23秒)。ページ記述言語にLIPS IVを搭載したことで，データ処理解像度が従来のLIPS IIIでの300dpiから600dpiに向上している。しかも，LIPS II+/IIIのエミュレーションモードでも600dpiでの出力が可能。さらに，ファインモードでスムージング処理を利用することで1200dpi相当(2400dpi相当×600dpi)の出力を実現した。用紙サイズはA3用紙までに対応している。標準に装備している給紙カセットで連続250枚の印刷が可能で，オプションカセットを2段追加することで最大850枚までの連続印刷ができる。ほかには，印刷時にトナーの消費量を半分カットして印刷する機能なども用意され，和文3書体，欧文12書体のフォントを内蔵している。インタフェイスはパラレルとシリアルが1系統ずつを装備し，オプションでもう1系統増やすことが可能。

価格は248,000円(税別)。

〈問い合わせ先〉
キヤノン(株)　☎03(3455)9320

バックアップ用電源 BX500 オムロン

BX500

オムロンは停電などの電源トラブルに対応するバックアップ電源「BX500」を発売した。

本機は出力容量が500VA(300W)で，定格負荷(500VA)のとき3.5分間，350VAのとき5分間以上のバックアップが可能。電源の切り替え時間は10ms，バッテリーの充電にかかる時間は8時間。バックアップ用の出力が2系統とスルー出力が1系統用意されている。また，停電，過負荷，トラブルなどがあるときは，LED表示とブザーで警報を発する。

大きさは285mm(幅)×287mm(奥行)×45mm(高さ)で，重さが5kgと従来機種「BX5」に比べて体積比30%減，重量比17%減を実現した。

価格は34,800円(税別)。

〈問い合わせ先〉
オムロン(株)☎045(411)7223，06(282)2672

調湿シート Keep Well 富士写真フイルム

富士写真フイルムは写真フィルムやビデオテープ，FD，MOなど記録メディアの長期保存に適した調湿シート「Keep Well」を発売した。

同製品は湿気による錆やカビ，変色，酢酸ガスなどによる劣化から各種記録メディアの品質を守る。効果は密閉状態で3年以上，湿度は30～40%RHの状態を維持し，交換時期は同製品の色の変化でわかる。大き

Keep Well

さは1枚あたり70mm(横)×90mm(縦)×2mm(厚さ)。

価格は1袋に5枚(1枚あたり4ピース)入って1,000円(税別)。

〈問い合わせ先〉
富士写真フイルム(株)　☎03(3406)2338

見えるラジオ RF-VR100 松下電器産業

RF-VR100

松下電器産業は見えるラジオ「RF-VR100」を発売した。

本機はFM文字多重放送をより正確に受信するために本体に収納可能なロッドアンテナを装備している。受信周波数はテレビが1～3ch，FMが76～90MHzで，最大12局までプリセット可能。音声出力はモノラルの小型スピーカを内蔵し，ステレオインサイドホンとのスイッチによる切り替えが可能。また，受信した文字多重データをパソコンに表示記録するためのデータポート(8ピン小型専用端子)も用意され，発売予定のインタフェイスキットを使用すればRS-232Cを通じてデータの転送が可能。送られてきた文字情報は最大238ページ分(1ページは15.5文字×2.5行)まで本体に記憶できる。

ほかには，電源がOFFの状態でもFM文字多重放送による緊急情報を監視し，地震，災害などの緊急情報番組受信時には警告音とともに文字情報を表示して自動記憶する緊急情報機能を内蔵している。

価格は27,800円(税別)。

〈問い合わせ先〉
松下電器産業(株)　☎06(909)1021

FILES Oh!X

このインデックスは，タイトル，注記——著者名，誌名，月号，ページで構成されています。食べ物が美味しい季節ですが食べ過ぎに注意して。

一般

▶NEWS
「WindowsWorld Expo Tokyo95」のレポートやIBMがロータスを買収したニュースなど。——編集部，ASAHIパソコン，7·15号，8-10pp.
▶特集1　DOS/Vマシンの選び方教えます
カタログの見方から新技術の解説，ショップブランドの紹介，全国のパソコンショップ情報まで，DOS/Vマシンの購入を助ける。——編集部ほか，ASAHIパソコン，7·15号，16-29pp.
▶98ユーザーのためのマッキントッシュ教室　17
今回はWindowsとMacintoshのAV機能をハードとソフトの面から比較する。——荻窪圭，ASAHIパソコン，7·15号，94-97pp.
▶特集2　新世代の大容量メディアを比較する
MD，PD，MO，Zipの4種類のメディアを速度やコストなど5つの観点から比較する。——本田雅一，ASAHIパソコン，7·15号，100-108pp.
▶GlobalInterface
「COMPUTEX TAIPEI'95」のレポート。——編集部，ASAHIパソコン，7·15号，118-121pp.
▶THE NEWS FILE
Windows95に対応したゲームソフト移植の話題やCGを使った新しいアニメーション制作技術の紹介など最新ニュース。——編集部，LOGIN，14号，48-53pp.
▶特集　パソコンの恐い話
パソコンを使っていて実際に起きた怖い話(失敗談)。——編集部，LOGIN，14号，147-161pp.
▶インターネットの心
インターネット接続講座や電子メール機能の説明など。——編集部，LOGIN，14号，184-187pp.
▶海外ゲーム最新事情
メーカー別の「E3」レポート。いち押しゲームの紹介もあり。——編集部，LOGIN，14号，192-199pp.
▶くねくね科学探検隊　第23回
インターネットの現状を紹介し，その可能性について考えていく。——鹿野司，LOGIN，14号，216-219pp.
▶こだわりゲーム年代記
今回はパソコンゲームに登場するオリジナルロボットからパソコンゲームの歴史的背景や魅力を探る。——与志田拓実，コンプティーク，8月号，100-101pp.
▶NEWS COLLECTORS
パナソニックの「REAL II」値下げや「東京おもちゃショー」で発表された「Pipinパワープレイヤー」の話題など。——編集部，電撃王，8月号，30-37pp.
▶第1特集　価格破壊で見えてきた新勢力図
SEGAの岡村氏とSCEの佐伯氏による対談，ウルトラ64のスペック予想，3DOの64ビット機など新世代ゲーム機の情報。——編集部，電撃王，8月号，30-45pp.
▶コンピュータが学べる学校紹介
全国にあるコンピュータやソフトを専門的に学べる学校を紹介する。——編集部，マイコンBASIC Magazine，8月号，31-40pp.
▶パソコンレクチャーX
PDの特長を漫画で簡単に紹介。——くりひろし，マイコンBASIC Magazine，8月号，72-76，182pp.
▶Arcade Game Graffiti　第18回
1983年に発売されたアーケードゲームの紹介。今回は「ハイパーオリンピック」「チャンピオンベースボール」「エレベーターアクション」など。——編集部，マイコンBASIC Magazine，8月号，164-167pp.
▶98ユーザーのためのマッキントッシュ教室　18
今回はPC-98とMacintoshのインタフェイスと周辺機器の違いについて解説する。——荻窪圭，ASAHIパソコン，8·1号，114-117pp.
▶特集2　台湾生まれの世界企業　エイサーの素顔
1994年世界第7位のパソコンメーカーとなったエイサーの秘密を探る。——編集部，ASAHIパソコン，8·1号，122-127pp.
▶特集3　PHSはパソコンの強い味方になるか？
PHSの解説やPHSを使ったパソコン通信の実現を紹介する。——編集部，ASAHIパソコン，8·1号，128-133pp.
▶3Dグラフィックスの世界へようこそ
3Dグラフィックに関する基本的な知識や作業の流れの簡単な解説とWindows用のソフト紹介。——野口修，I/O，8月号，17-22pp.
▶MultiMedia Watching　20
「E3」やデジタルデータ放送サービス，移動体通信など，マルチメディアに関する話題。——奥野雅之，I/O，8月号，43-45pp.
▶ペンティアムはなぜ速いか
Pentiumプロセッサのアーキテクチャを徹底解説する。——編集部，I/O，8月号，57-60pp.
▶Pentium vs 486
Pentiumとi486系のCPUの違いを解説する。——田嶋孝行，I/O，8月号，61-63pp.
▶インターネットアクセスガイド　3
wwwの紹介。今回は関東地方の通信・放送事業者やプロバイダのwww。——森羅万象，I/O，8月号，110-111pp.
▶DeskTopMusic入門　5
今回は複数の楽器を組み合わせた入力方法を紹介する。使用ソフトは「レコンポーザ」。——あまだたかし，I/O，8月号，123-127pp.
▶特集1　夏期ニューマシンオリンピック
機能別に購入条件を考えた新型マシン紹介。——編集部，ASCII，8月号，269-306pp.
▶インターネット膝栗毛　ROUTE 6
今回はインターネットで本を買うことをテーマにいろんな本屋を探してみる。——編集部，ASCII，8月号，373-380pp.
▶Digital Beat Zoo
ヤマハのXG音源「MU50」を紹介し，各社の最新音源を比較する。——編集部，ASCII，8月号，385-388pp.
▶脳型コンピュータを作る　第1回
今回は脳の仕組みを知るために脳がどんなコンピュータなのか？　ということについて説明する。——編集部，ASCII，8月号，389-396pp.
▶'95東京おもちゃショーレポート
新世代ゲーム機の新作ソフトや子供用電子手帳などショーで目だったアイテムを紹介。——編集部，ASCII，8月号，432-433pp.
▶帰ってきたバカパパ BUPPiN
ゲーム環境を強化するアンプやスピーカ，プロジェクタなどのアイテムを紹介する。——編集部，ASCII，8月号，434-435pp.
▶特集　1995夏　コンフィグ大展覧会
MS-DOSの各種マシンにあらかじめ入っているコンフィグや各種ゲームをプレイするためのコンフィグの例を紹介する。——編集部，LOGIN，14号，147-161pp.
▶インターネットの心
3Dチャットが楽しめる「World Chat」の紹介や初心者用接続講座。——編集部，LOGIN，14号，184-187pp.
▶ゲーム大国　その名は台湾
「COMPUTEX TAIPEI'95」のレポートと台湾のパソコンゲーム市場の動向を紹介。——編集部，LOGIN，14号，192-197pp.
▶スペクトラムホロバイトが来た！
フライトシミュレータで有名な同社へのインタビュー。——編集部，LOGIN，14号，198-201pp.
▶くねくね科学探検隊　第24回
今回は1カンデラの青色高輝度ダイオードを開発した話。——鹿野司，LOGIN，14号，216-219pp.

X1/turbo/Z

X1turboシリーズ
▶KAIJYUU
空から落ちてくる怪獣をやっつける。——青山正実，マイコンBASIC Magazine，8月号，115-116pp.

X68000

▶SUPER SOFT INDEX
新作ソフトの予定表。X68000用は「EXCITINGみるく」など。——編集部，コンプティーク，8月号，125p.
▶電撃新作予定表
機種別の予定表。X68000用は「プリンセスメーカー」が発売予定。——編集部，電撃王，8月号，180p.
▶MAGIC BALL SHOOTERS
悪の組織に占領された島を取り戻す，リアルタイムシミュレーションゲーム。——パト，マイコンBASIC Magazine，8月号，117-121pp.
▶FINAL FANTASY VI～オペラのアリア～
NAGDRV＋GS音源用の音楽プログラム。——渋川大吾，マイコンBASIC Magazine，8月号，126-127pp.
▶SUPER SOFT Hot Information
X68000用は予定表のみでシューティングゲーム「ジャンクドライブ」など。——編集部，マイコンBASIC Magazine，8月号，とじ込み付録11p.
▶売れ筋ハード＆ソフトBEST20
1995年5月期の九十九電機でのハードとソフトのランキング。ハードではX68000 CompactXVI，ソフトでは「シャーペンワープロパック」「CD-ROMドライバ」などが登場。——編集部，ASCII，8月号，260-261pp.
▶ONLINE SOFTWARE INDEX
大手ネットにアップロードされたプログラムを紹介する。X68000用はSCSIデバイスの属性をチェックする「AskSCSI.x」。——編集部，ASCII，8月号，500-501pp.
▶SX-WINDOWプログラミング　第22回
今回はxgccを使ってプログラムを作成する場合によくあるトラブルや疑問点について答える。——吉野智興，C MAGAZINE，8月号，126-131pp.

ポケコン

PC-E500
▶4 BLOCK BATTLE
ブロック落とし風4ならべ。——石川大介，マイコンBASIC Magazine，8月号，122-123pp.

参考文献
I/O　工学社
ASAHIパソコン　朝日新聞社
ASCII　アスキー
コンプティーク　角川書店
C MAGAZINE　ソフトバンク
電撃王　主婦の友社
マイコンBASIC Magazine　電波新聞社
LOGIN　アスキー

QUESTION and ANSWER

Oh!X 質問箱

6月号のローテクでミキシングがありましたけど，電源を12Vから9Vにする理由がわかりません。私は12Vをパソコンから取って使ってます。　福島県　高田　英夫

NJM4580というオペアンプは，プラスマイナスの電源を与えても単電源でも動くオペアンプです。

記事中ではこのオペアンプを，12Vのアダプタから取り，9Vレベルまでレギュレータを使って落としています。また，電源はこれひとつ，つまり単電源回路です。

高田さんがいうとおり，本体の12Vから電源を取ってもオペアンプは駆動されます。もちろん，アダプタから12Vを直接とっても駆動しますが，コンピュータ本体から流れる電源は，それほど大きな出力を取ることはできませんし，デジタル回路のノイズがたんまり乗っています。アダプタのほうも，普通の安いアダプタの場合だと内蔵でレギュレータやツェナーが入っていることは希なので出力電圧は多少リップルが残り，落ち着いていません。

オペアンプを利用したアナログ回路では，供給電源の安定さが出力に影響することがままあります。そのため，レギュレータを使って電源を安定化させたのです。

いま，ウィンドウマネージャを制作していますが，64k色でやっているので，malloc()で65535バイト以上確保すると止まってしまいます。-z-heap以外に，なにかよい方法はないでしょうか。いい忘れていましたが，言語はgccです。　埼玉県　佐々木純一

mallocはヒープ領域からメモリを確保します。おっしゃるとおり，-z-heap:を利用すれば，ヒープ領域を拡張できるので，ちゃんと64Kバイト以上メモリを確保できます。x86系のCコンパイラと違って，この64Kバイトというのは，OSやCPUの問題で起きるわけではなく，単なる「デフォルト」の値が64Kバイトであるだけです。だから，ヒープ領域を拡張することを宣言すれば，それで話は済みます。

それには，リストのようにいちばん頭に，allmem()の一文を入れます。このallmem()は，空いているすべてのメモリをヒープとして解放する命令です。この宣言をしておくことにより，mallocを利用するとき，いくらでもメモリを確保することができます。リスト中，rstmem()はallmem()の解放です。

ただし，allmem()，rstmem()はローカルな関数であるため，これがあると移植性に問題が起きます（ハードウェアべったりなプログラムを作ってるようなので，気にすることはないでしょうが……）。

方法はもうひとつあります。

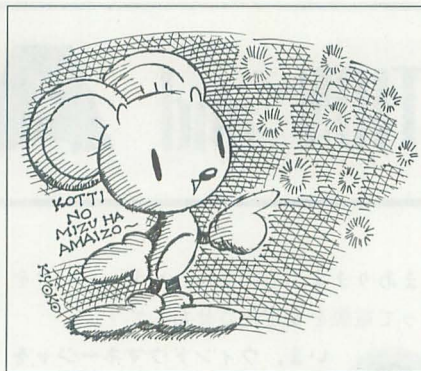
せっかくgccを利用しているんですから，libcを利用すればいいのです。libcのmallocは，自動的にヒープを拡張するので，allmem()，rstmem()は必要ありません。ただ，libcはUNIX系の標準ライブラリがきちんと揃っているので便利ですが，X68000でゲームなどを作ろうとする場合には，いささかライブラリが不足するかもしれません。もっとも，XCのライブラリがゲームに使えるのか？　というと，そうともいい切れない部分もあるので，どうしても標準環境でなくてはイヤとか，ライブラリがアセンブラコードで書かれてないとデカくなるからイヤとかいう主義があってXCを使いたいというのでないならlibcを利用することをおすすめします。　（瀧　康史）

リスト1

```
#include        <stdio.h>
#include        <stdlib.h>

#define         MEMSIZE         1024*1024         /* 1MB  */

void    main( void )
{
        void    *data_pointer;                  /* メモリ確保先のポインタ */

        if(allmem() != 0){                      /* すべてのメモリを利用する宣言 */
                printf("allmemでメモリの確保に失敗しました¥n");
                exit(-1);
        }

                if((data_pointer=malloc(MEMSIZE))==0){
                                                /* 実際にはmallocでメモリを確保する */
                        printf("mallocでメモリの確保に失敗しました¥n");
                        exit(-1);
                }
                free( data_pointer );           /* malloc確保されたメモリの解放 */
        rstmem();                                               /* メモリの解放 */
}
```

質問にお答えします

日ごろ疑問に思っていること，どんなことでも結構です。どんどんお便りください。難問，奇問，編集室が総力を挙げてお答えいたします。ただし，お寄せいただいているものの中には，マニュアルを読めばすぐに解答が得られるようなものも多々あります。最低限，マニュアルは熟読しておきましょう。質問はなるべく具体的に機種名，システム構成，必要なら図も入れてこと細かに書いてください。また，返信用切手同封の質問をよく受けますが，原則として，質問には本誌上でお答えすることになっていますのでご了承ください。なお，質問の内容について，直接問い合わせることもありますので電話番号も明記してください。

宛先：〒103　東京都中央区日本橋浜町
3-42-3
ソフトバンク株式会社出版部
Oh!X編集部「Oh!X質問箱」係

FROM READERS TO THE EDITOR

まだまだ残暑厳しく，つらい日々が続きますが，確実に季節の移り変わりを肌で感じることができる時期がやってきました。この季節，食べ物がおいしいからといって，夏の間に失った体力を取り戻そうとしての暴飲暴食は控えましょうね。

◆7月号の特集は絶品でした。こういう技があるからアセンブラは魅力的なんですよね。ただ，個人的にC言語を理解できないだけという説がありますが。とにかく世界で最後になるかもしれない「すべてがユーザーに開放されたパソコン」を扱う雑誌にふさわしい内容でした。

三宅　涼(17)京都府

◆7月号の特集にあった西川善司氏の「一線を超えた68系プログラマ養成講座」は，楽しく読ませてもらいました。ほぼ独学の私としては，知らなかったこともあったり，当然のように使っているものもあり，ニヤニヤしながら読ませてもらいました。

猪狩　友則(21)千葉県

◆7月号の特集は，久しぶりにアンケートハガキを書く気にさせてくれました。やはり西川氏の記事に気合を感じます。こういった記事は，ほかの雑誌ではあまり見られない(68系にかぎれば皆無でしょう)ので，Oh!Xにはがんばってもらいたいところです。この力が果たして次世代X機で爆発するのでしょうか。それ以前に爆発できるのでしょうか?!

片桐　健喜(26)愛知県

◆自分はアセンブラはまったくできず，Cだけの人だが，7月号の特集を読んでちょっと燃えてきてしまった。う～ん，美しく速いアセンブラへの挑戦。うぉー！　時間をくれえ！

鈴木　達也(21)愛知県

初心者にはつらい，という声の聞かれた7月号の特集。難しいからとあきらめずに挑戦してみましょう。時間はかかるかもしれませんが，きっと得るものがあると思いますよ。

◆7月号37ページのコラムを読んで「オールマシン語」以上のステータスであった「ダイレクトコーディング」は，いまどこへいってしまったのでしょうか。私が中学生のとき使っていたのは，ポケコンだったのでアセンブラのような贅沢品は使えませんでした。それにしても「魔神語」……懐かしい文字，あの頃がんばっていた人たちは，いまなにをしているのでしょうか。おとなしくMS-DOSの軍門に下るとは思えないし。

高須賀　義弘(19)香川県

そうですね。8ビット機全盛だった，僕の高校時代の先輩にもダンプリストが読める人がいました。いまごろなにをしているのだろうか。

◆プログラムの最適化は面白いですよね。自分のちょっとした工夫で，何倍も実行速度が速くなることもありますからね。でも，それに没頭してプログラムの全体像が，なかなかできあがらないという事態もよく起きることです。ほどほどが肝心ということでしょう。

中島　康弘(36)群馬県

最適化のつもりがエンバグしたりすることもありますからね。世の中うまくいかないものです(そういう問題じゃないって)。

◆私は，コンパイラの話を聞いたあとから，プログラムを無理して書かなくなった。ある程度のところまで書いておけば，あとはコンパイラが勝手に最適化してくれるというのだから。でも，やはり無駄なものは滅多に残しておきません。これは性格の問題でしょう。

中村　学(23)福岡県

コードの美しさを求めるのも，またひとつの最適化かもしれませんね。

◆「初心者向けの記事を」という意見について。Oh!Xは「雑誌」よりは「専門誌」という感じが強い。いま，初めてOh!Xを手に取った人がいたとして，どれだけ読みこなせるのだろうか。こんなことをいうと「そういう人は，それなりにX68000に興味をもっているから大丈夫」というとても悲しい答えが返ってくるかもしれない。つまり，現状でのX68000の存在価値を知らないかぎり，Oh!Xは受け入れられないことになる。確かに真のX68000ユーザーにとっては魅力ある雑誌であるが，初心者は初心者なりにX68000を楽しみたいのである。具体的には，やはり流行のウィンドウ(SX-WINDOW)から始める，ということに落ち着くでしょう。毎月SXツールを使いながら，気軽にX68000を楽しめるコーナーなどがあれば，初めての人にも読みやすいのではないでしょう。

上池　宏幸(20)福井県

◆「初心者にやさしいOh!X」という言葉がよく聞かれますが，私は別にこのままでよいと思う。しかし，少しでも理解できるように7月号の特集にあった西川氏の記事みたいに，小さいコラムを作って難しい概念などを詳しく説明してほしい(たとえば，クロックってなんだ？　とか，*.Xファイルの謎シリーズ，とか)。そうすれば初心者の人たちも難しい辞書を引いて悩まずにすむと思います(悩んだほうが勉強になるかもしれませんが)。

中村　慶彦(16)山口県

「初心者向けの記事を」という声に，このほかにもいろいろな意見が届きました。どこまでやれるかわかりませんが，できるだけ読者の皆さんの要望に応えるべく，がんばりたいものです。

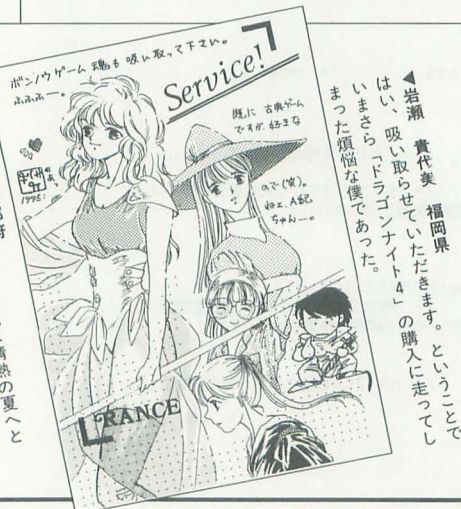
◆以前紹介されていた瀧氏の黒塗りのCM-64を思い出して，我こそもと思い色を塗ってしまったが，とてもツヤのある漆塗りのようになってしまった。ますます本体(X68000 XVI)との距離が開いた。

杉浦　晃規(20)愛知県

こうなったら，X68000 XVIをCM-64のように塗り変えてしまいましょう。そうすれば，バッチリさ(同じように失敗しそうだけどね)。

◆「フリーソフト分類検索95」を眺めていて「X

◀岡村　直也　京都府
長かった春に終止符を打ち，一気に情熱の夏へと爆進する2人(笑)。最後のコメントが気になるけど，とりあえずよかったね。恵子ちゃん。

◀岩瀬　貴代美　福岡県
はい，吸い取らせていただきます。ということでいまさら「ドラゴンナイト4」の購入に走ってしまった煩悩な僕であった。

68000用もいっぱいあるな……おっよさそうなものがあるぞ。ほしい」と思っていたところへ，THE USER'S WORKS SPECIALにSX CALCの記事を見つけました。静止画であっても実際の画面を見られるのは，非常に参考になりとてもありがたいです。特集でフリーソフトの使用報告や評価のようなものをやってもらえると，すごく嬉しい人(私も含めて)が多いのでは？

秋山　義豊(36)高知県

今月からTAKERU関係の同人ソフトを紹介するページができました。また，ゲームに限らず，いろいろ取り上げようと考えていますので期待してください。

◆SCSIボードを買った。ハードディスクをつなぐのは当然として，その次のリムーバブルメディアになにを選ぶべきか非常に悩む。MO, PD, Zip, MDなどいろいろなメディアがありすぎ，また進歩(主に容量の増加)が激しすぎる。

暮石　徹(23)奈良県

単一機種で停滞した状況より，競争相手が出ていろいろ動いているほうが面白いんですけどね。まあ，X68000でいますぐに使うのであれば，やはりMOかな。

◆念願のMOドライブをついに買いました。購入後，家に帰って買っておいたOh!Xを読むと私が買った値段とほぼ同じ定価で，キャッシュ容量が倍のMOドライブが発売になっていました。ショーック！

津村　忠蔵(20)佐賀県

買い置きはいけません。今度からは，きちっと情報収集してから買い物にでかけましょう。

◆30ピンSIMMを中古で1枚1,000円で見つけました。これは安いと思い，さっそく購入して家に帰り，装着して起動してみる。試しにRAMディスクを多く確保して，データをコピーしてみると……ゲゲッ，データが化けてる！　メモリチェックをしてみるとアドレスエラーが(ひぇ〜)。皆さんも安いメモリには注意してくださいね。

千葉　幸雄(19)東京都

うまい話にゃ裏がある。今度からは気をつけなきゃね。

◆THE USER'S WORKS SPECIALで紹介されていた「クイズジョッキー」をTAKERUで入手し，プレイしてみました。かなり面白かったと思います。問題はちょっと理系的でした。

一ノ瀬　宣彦(24)群馬県

クイズ番組形式という統一された世界観，それにゲームとしての作りが，きちんとなされているのもポイント高し！　というところですか。

◆最近になって，フリーソフトを多用するようになりました。そのせいあってか，コマンドや設定などに対する恐怖心のようなものがなくなりつつあります。こうやって少しずついろんなことを覚えていくのかな，と思うとなんだか嬉しいものです。自分らしい環境を作り上げるということが，たいへんな分だけ楽しいことだと気づくことができました。近々，Xellent30をつけてあげるから，もうひとがんばりよろしくな。

◀占部　哲彦　広島県
9月号とはいえ，このコメントを書いている現在は夏真っ盛り。こういう涼しげなイラストを見ると心が落ち着きます。

◀佐藤　貴是　神奈川県
机にかじりついて技術を磨くのもいいのですが，たまにはネタを探していろいろ行動してみましょう。面白いイラストを待っていますよ。

XVI殿。

和田　哲也(25)東京都

パソコンを使う楽しさを見つけることができて，本当によかったですね。

◆CD-ROMドライブをついに購入しました。1月号にあったプログラムでPhoto CDもOKです。音楽も聞けるし，なかなかいい買い物をしたと思っています。

狐塚　一浩(21)栃木県

対して，僕の買ったCD-ROMはというと……う〜む，ただのシネパックプレイヤーとしてしか機能していない気がする。ちょっともったいないかな。

◆DSPボード「AWESOME-X」は，まだ発売されないのでしょうか。早く紹介記事が載るといいのですが。あと，これを機会に各DSPチップの紹介を特集でやってもらえないでしょうか。どのクラスのDSPなら最近のCPUより高速なのでしょうか。

下川　将紀(24)東京都

8月号では簡単な紹介，そして今月号から本格的に活用するための講座が始まりました。ぜひ感想をお聞かせくださいね。

◆そういえば，ずっとX68000を使っていたから気づかなかったけれど，次の機種でスプライトやBGが搭載されるとはかぎらないんだよな。う〜ん，テーマ曲もないだろうな。それ以前に新機種が出るかが問題だ。

新谷　貴幸(20)埼玉県

時間はかかってもきっと出してくれるでしょう。メーカーを信じてあげなくちゃ。

◆僕の夢のパソコン，それはPC-8801MA2でした。そして，ついに手に入れたのです。ゴミ捨て場で……嬉しいやら悲しいやら。でも，PC-8801MA2のためにモニタ切り替え機まで買ったし，あとは「スナッチャー」と「ジーザス」さえあれば……。

永井　孝(20)神奈川県

人生バラ色ですか？

◆書店で親子がPC-98の本を探していた。「どこ？」「あのえっくすろっぴゃくはちじゅうの横」……2桁も足りないぞ。

菰田　英和(25)奈良県

確かにたいていい間違えるときには，桁数が不足していますね。たまには「えっくすろくじゅうはちまん」といい間違える人がいてもいいだろうに。

◆任天堂が以前発売した「スターフォックス」や「Xウィング」なんかより，ずっと昔にジャレコから発売された「エクセリオン」のほうが，すごいゲームだと思うのですが。あと，ハガキの内容を全部読んでいることに確信がもてたので，うかつなことは書かないようにしたいと思った。

佐久間　利弘(18)千葉県

人間，正直なのが一番です。猫をかぶらず，思ったことをガンガンアンケートハガキにぶつけましょう。あまりにヤバすぎる内容だと困っちゃいますけどね。

◆この前，東京に奨学金のための面接を受けにいった。4人で面接をしたのだが，1人だけ面接官の笑いを取ることに終始してしまった。

山脇　彰(20)福岡県

不真面目な人間ととられるか，それともユーモアのある人間ととられるか……無事合格することを祈っています。

◆X68000を購入した1年前には，X68000でゲームをする気にはなれなかった。しかし，初夏のある日曜日，気がついたら手には「ファンタジーゾーン」と「ぷたさん」「リブルラブル」「バラデューク」そして「悪魔城ドラキュラ」があった。後悔はしていないのだが，本業をする暇がない(現在，人形の館でハマッてる)。

師　茂樹(23)東京都

ゲームは本業の合間にやるものですが，なんて偉そうなことをいっていますが，本業の合間に「キングスフィールドII」にハマッている僕なのでした。ま，人間たまには息抜きしなきゃ。

◆髪を切って，ただいま電子ちゃん状態。……ぐしぐし。

岩瀬　貴代美(23)福岡県

それは，電子ちゃんみたいな性格だと思われるのが嫌だからなのかなあ。そんなことはないですよね。

◆会社の定期診断にいってきました。食費を削っているおかげで，体重が7.6kgも減りました。あと2kg減らせばなんとかカラープリンタが買えそうです。

久米　豊信(26)群馬県

かわいそうに，久米さんはX68000という魔物にとりつかれてしまったのですね。くわばらくわばら。

◆今年も就職が厳しいのですが，僕や友人たち

◀岩本　理博　兵庫県
きっとCD-ROMの購入が先になると思います……。もちろん，がんばって制作しているようなので発売を期待して待っていましょう。

は結構気楽にやっています。なんたって，就職が決定していなくても“当たり前”という空気がそこらじゅうにいっぱい(笑)。プレッシャーがないというのはいいものですな。

今井　健生(23)奈良県

のんびりしている間に仲間からひとり，またひとりと内定者が出て，最後に残ったのが今井さんだったら……(不安を煽ってどうする)。

◆7月号のアンケートハガキが，縦横逆にくっついていて，キリトリ線が無意味となっていて，切り取るためにはテクニックが必要だった。この頃のOh!Xの内容もさることながら，アンケートハガキの切り取りも上級者向けになるとは…….

北田　顕啓(22)北海道

さらに，65536冊に1冊あるというプロフェッショナル向けのハガキもあるとか(大ウソ)。

◆この間，うちの会社にソフトバンクの重役の方がきて講演されました。“マルチメディアネットワーク”という題でした。それにしても1人当たり2.8台のパソコンをもち，「ロータスノーツ」とネットワークでいろいろな情報が引き出せるなんてさすがですね。

篠塚　哲(26)神奈川県

ふうん，そうなのか……(ってあんたは社員だろうが!)。

◆そういや僕も最近腰が痛いな。ANOTHER CG WORLDはタイムリーだったかもしれない。この歳でぎっくり腰は，ちょっとかっこ悪い。以後気をつけよう。ところで，他人にどのパソコンを勧めるべきか困ってしまうのは，みんな同じようですね。

清水　弘和(18)東京都

ぎっくり腰にはなったことがありませんが，最近，体力もなくなっているしちょっとだけ心配かな。

◆うちの姉に長男が生まれました(本当)。名前は「ゆうと」。ユートピアのユートらしい。ちなみに某コメディアンとは関係ないらしい。

岡村　直也(24)京都府

それはおめでとうございます。でも，岡村さん，ゆうと君をさっそく4コマのネタにしようとか考えていません?

◆某家庭用ゲーム機でプログラムを組んでいる人間はよくわかっていると思うが，スプライトは大きければ大きいほどよく，定義数も画面を埋めつくせる以上にあったほうがいい。それにハードのスピードに任せてガリガリとポリゴンを描くよりは，V-BLANKに合わせてスプライトのデータをチョロッと転送するほうが私は好きだ。最近DOS/Vを買って少しプログラミングをしてみたが，V-BLANKの割り込みがなくて悲しい。次期Xにはそうなってほしくない。

千葉　浩貴(22)宮城県

仮にDOS/Vのようなマシンになってしまったら，それはDOS/Vクローンであって，Xシリーズと認められないでしょう。シャープさん，がんばってね。

◆自宅から某宗教団体の本部までさほど遠くないので，周りをヘリコプターが飛び回り，道にはパトカーがウロチョロ。別に悪いことをしてなくてもコソコソしちゃうのはなぜ?

山口　文隆(21)静岡県

きっとそれは，普段の生活態度に自信がないからですよ。悪いことをしていないのなら堂々としていればいいんです(すごく偉そう)。

◆「ときめきメモリアル」をプレイしていて，自分が高校時代，いかに無味乾燥な学生生活を送っていたかということを悟らされた(いやマジで)。

藤田　康一(24)静岡県

でも，後悔はしていないんでしょ。だったら問題はないでしょう。

◆「相棒――一緒に仕事をする相手。また，仲間」なるほど，江口さんが「響子inCGわーるど」で書かれていた「X68000はツールというより相棒みたいなもの」という考えは，私も同感です。きっと68ユーザーの多くは，同じ考えをもっていると思います。だから現在のような厳しい環境になってもなお，X68000にこだわり続けるユーザーがたくさんいるのだと思います。大切な相棒を裏切ることはできませんからね。いつもなにかを考えさせてくれる「響子inCGわーるど」も，もう50回を迎えられたのですね。おめでとうございます。体を一番大切にしてこれからもがんばってくださいね。

加藤　和人(19)愛媛県

そうですね。たとえほかのマシンを使うことになっても，きっとしばらくは手放さないでしょう。さすがに5年も使っていれば愛着もわくというものです。

◆就職活動の最終面接にて。「どんなパソコンを使ってるの?」「MS-DOSは使ってる?」「WINDOWSは?」……やはり避けて通れないこれらの問いに，X68000ユーザーの宿命を感じました。しかも相手は某国民機を日本一売りまくっているコンピュータ販売会社の社長。「学校ではよく某国民機を使っています」とフォローをいれたけど，だめだなこりゃ。でもK女子大4年のM子さんに「お互いがんばりましょうね」なあんていわれた日にゃ，もうがんばるしかないよな。うんうん。

大畑　佳史(21)兵庫県

頭ごなしにそういういい方をされると，ち

第36回(?)
BEAT君(X68000 ACE-HD改)復活への道!

いま思い起こせば私は彼(BEAT君)にずいぶんとむちゃなことをしてきました。

1)　無理やりビデオ出力(しかも15MHzで)
2)　無理やり3.5″FDD内蔵(どこに?)
3)　ジョイスティック6ボタン←→2ボタン変換スイッチ内蔵(1,2両ポート)
4)　もちろんクロックアップ(大失敗)

などなど。ビデオ出力は，ジャンクのメガドライブのチップを流用して，いい加減に組み立ててしまったもので，はっきりいって使いものにならなかった(映ることは映るけど)。

3.5″FDDの内蔵は，もう超変態技を使ってしまった。でも，途中でFDD周りの基板を1枚おしゃかにしちまった(12,000円なり)。

6ボタン←→2ボタン変換スイッチは，セガパッドをばらして配線し直せば簡単だったのだろうけど，なぜか彼をいじめるようなことをしてしまった。

そして，BEAT君最大の危機と思われたクロックアップ。動け!　動くんだ～と3日間彼とバトルを繰り広げたものです。そんな彼が1994年9月8日の雷に打たれてポアしちまうなんて信じられませんでした。もうすぐ1年経ってしまいます。メイン，サブ基板は，ともにピンピンしています(動作確認ずみ)。電源がどうにも直らないんです。秋葉にパーツを買いに行きたいのですが，本当に暇がないっす。誰か電源だけを安く売ってくれませんかねえ。

千装　茂夫　埼玉県

いま明かされるBEAT君の衝撃の過去(笑)。本当にむちゃをしていたんですね。忘れないうちに，がんばって復活させましょう。でも，ネタにしてくるということは，ちゃんと気にかけていることだから大丈夫なのかな。

ょっとムッとしちゃうかな。ちゃんと自分のやってきたことを主張できるならまだいいけどね。それにしても，就職活動がたいへんなのか，たんなるのろけ話なのか，よくわからないけどとりあえずがんばろう。

◆私はいま，パソコンショップでバイトをしています。しかし，現在のパソコンの無個性なことには呆れ果てます。メーカーの初心者を食い物にしたような売り方にも困ったものです。シャープには，ぜひともNEW Xを出して業界に新風を……といいたいですが，PC-8901なんか出しているようじゃね。　山中　祐司(19)栃木県

最近の風潮は僕も気に入りませんが，それでもきっかけはどうであれ，パソコンを使う楽しさを感じてもらえればいいんじゃないか。そう思うようにしています。

◆「CARD PRO-68K」を使って所有しているX68000用のソフトウェア管理をしているのですが,手つかずのゲーム(買ってきて少し遊んでみただけ)が,約半分を占めていることがわかります。X68000用の新作ソフトがほとんどない現在，このフロッピーケースの中に埋もれているゲームで遊びながら，次期Xシリーズの登場を待ってみたいと思います。ちょっと消極的かな。

後迫　浩一(34)神奈川県

改めて遊ぶことで，購入当時は気づかなかった新しい発見があるでしょうから，消極的な行動でも，無駄なことでもないと思いますよ。

◆本屋でMS-DOS ver.3.3の本とコンピュータ用語辞典を買った。MS-DOSの本を買ってから，Human68kがわかってきて，自分でバッチファイルなどが作れるようになり，やっと1993年10月号の付録ディスクの「チェリーボーイ」が動いた(以前は常駐の意味がわからなかった)。そして辞典を買ってからOh!Xに出てくる専門用語がわかってきて，面白さが倍増した。そして，その辞典にはソフトバンクの社長まで載っていた。

片山　明義(17)奈良県

うん，片山さんはいい辞書を購入しましたね(一応社員ですから。なんてね)。

◆「Griffon」を遊ぶためにX1turboZIIを用意して待っています。完成したら申込方法を掲載してください。　滝井　浩明(26)兵庫県

結構期待している人は多いですよ。BI-Factoryさん。がんばって完成させてね。

◆少し前のことですが，付録ディスクに「SION IV」のデモが入っていましたが，完成したのですか。もしくはいつ頃出るのか教えてください。

渉里　健介(20)北海道

そういえば，すでに2年が経過していたんですね。制作は怪調に進んでいるといったところでしょうか。最近では、ポチポチ面データも作ってたりします。ま，気長に待っててください。

◆いまになって，MIDIボードと「ローテク工作実験室」(1994年9月号)にあった部品を買い漁ろうとしています。そこで，ウェーブブラスターがバージョンIIになっているのに気がついた。調べてみるとGS/GM/MT互換でエフェクトも内蔵しているらしい。はたして，インタフェイスは記事どおりに製作すればいいのか？　私のMIDI化計画に明日はあるのか？　というわけで，瀧氏のアドバイスを心待ちにするローテク工作若葉マークが富山にいたりする。

藤田　和久(32)富山県

バージョンIIでもそのままでOKです(瀧)。

◆TS-6BS1mkII情報Part3。以前「SCSIで書き込みエラーを起こすことがある」ということを書きましたが，ツクモさんにTS-6BS1mkIIを送れば，Xellent30sに対応できるようにアップグレードしてもらえます(原因は，私のSCSI装置とボードの相性が悪かっただけでした)。

福知　健(24)京都府

原因がわかったところでひと安心。ちょっとした細かいところで，動作が不安定になってしまうものですね。

◆先日，会社の先輩からいらなくなったOh!MZをもらいました。私はOh!Xに名前が変わったと同時に買い始めたので，X68000の発売当初のことをまったく知りませんでした。あのスペックに対する胸の高なりは，もう味わえないのでしょうか。New Xに期待します。

今井　敏弘(24)香川県

皆の熱い思いがメーカーに届くといいな。

▲澤田　友伸　兵庫県
確かに似ている「マッドストーカー」と「G2」。どちらもアクションゲームとしての爽快感を味わえる気持ちのいい作品ですけどね。

ぼくらの掲示板

●掲載ご希望の方は，官製ハガキに項目(売る・買う・氏名・年齢・連絡方法……)を明記してお申し込みください。
●ソフトの売買，交換については，いっさい掲載できません。
●取り引きについては当編集部では責任を負いかねます。
●応募者多数の場合，掲載できない場合もあります。
●紹介を希望されるサークルは必ず会誌の見本を送ってください。

売ります

★X68030用内蔵ハードディスクドライブ「CZ-5HI6」(160Mバイト)を50,000円以下で売ります。またX68000 XVI用内蔵メモリモジュール「CZ-6BE2B」を20,000円以下で売ります。連絡は往復ハガキでお願いします。〒535　大阪府大阪市旭区大宮4-6-30 3階13号室　染川様方　河田裕康(23)

★X68000 XVI用内蔵メモリモジュール「CZ-6BE2B」を20,000円前後で売ります(箱，説明書，すべてあり)。連絡は往復ハガキでお願いします。〒533　大阪府大阪市東淀川区東中島1-10-15 サニーハイツ新大阪1104号　宮崎　早夫(42)

★1Gバイトハードディスクを35,000円，540Mバイトハードディスクを15,000円，128/230MバイトMOドライブを50,000円で売ります。ハードディスクは箱，説明書なしのドライブのみ。MOは箱，説明書，付属品ありです。連絡は官製ハガキにてお願いします。〒457　愛知県名古屋市南区中割町4-89　県営中割住宅404号　神野　力(20)

★アイテック製ハードディスク(SCSI，240Mバイト)を送料込みで18,000円以上で売ります。箱，説明書，ターミネータなし。ケーブルはフル・ハーフ，ハーフ・ハーフのどちらか希望するもの1本をつけます。また，シャープ製24ドット熱転写カラー漢字プリンタ「CZ-8PC3」を送料込みで8,000円以上で売ります(箱，説明書付属品すべてあり)。連絡は往復ハガキでお願いします。〒706　岡山県玉野市和田3-24-21　野崎　国彦(21)

買います

★I・Oデータ製拡張スロット用4Mバイトメモリボード「PIO-6BE-4ME」を10,000円以上で買います。連絡は往復ハガキでお願いします。〒520-05　滋賀県滋賀郡志賀町小野朝日1-4-9　倉谷　圭(23)

編集室から from E・D・I・T・O・R

DRIVE ON

このコーナーでは，本誌年間モニタの方々のご意見を紹介しています。今月は７月号の内容に関するレポートです。

●７月号の113ページのハミダシに「以前のようなプログラミング重視の内容希望」というのがありました。偶然の一致でしょうが，今回の特集はその希望に沿っているのではないかと思います。

私はZ80以外のCPUとコミュニケーションが取れない人ですが，MC68000がZ80なみに安くなってきたいま，使えるようにならないと大損だという気持ちが強くなってきました。そういう意味でもかなり興味深く読ませていただきました。最適化というきわめて楽しい作業にも定石があり，これを継承していく必要性も機会もアマチュアレベルではなくなってきている現在，Oh!Xだからこそできた内容だと，読んでいて非常に爽快でした。

また，C言語，それもGCCについての内容はX68000ユーザー以外の多くのUNIXユーザーにも関心の高いものであるはずですが，この記事のことをX68000ユーザー以外はほとんど知らないというのが残念でなりません。

現在，仕事でC言語を使っていますが，最適化(高速化)が大きなテーマになっています。それだけにこの内容はあまりにもタイムリーでした。直接役に立つことはなくても，その内容に接点があれば，いままで見えなかったものが見えてくるようになるからです。

考えてみれば，Oh!MZの頃からプログラム指向の強い雑誌でした。しかも莫大なリソースを消費することをよしとしない風潮があり，長い時間を経たいまも，多くのユーザーや開発者が忘れようとしているものをいまだにもち続けていることに嬉しくなりました。

浅野　憲(24)　X68000PRO, Macintosh Centris650, PC-98RL, X1F, AppleII, PC-1600K, PC-1245　東京都

●プログラムの最適化というと，アルゴリズムの考案，選択やプログラム全体の構成などに目がいってしまいがちですが，今回の特集「Optimizing Method」では思いっきり小手先の技を披露してくれました。こうしたテクニック関係は機種やCPUに依存しがちなので，１冊の本にまとまることが少ないんですよね。というわけで，なかなかありがたい記事でした。

今後は，グラフィック関係，サウンド関係，ファイルアクセス関係のテクニックとシリーズ化して，本当に１冊の本にまとめてもらえると嬉しいです。

中村　健(25)　X68000 ACE, PC-386GS　埼玉県

●「THE USER'S WORKS SPECIAL」を見て，X68000はまだまだ元気だなぁ，と感じました。実態はつかめませんが，X68000はユーザー対プログラミング人口比が日本一の機種ではないかと思っています。ここにあるどのソフトを見ても，ひと昔前の同人ソフトより質が高そうに感じました。私が気に入ったのは「クイズジョッキー」と「SX CALC」です。

ソフトハウスから発売されるソフトが圧倒的に少ないいま，X68000ユーザーがこのようなソフトに拠りどころをみつけるのもひとつの手だと思います。市販のソフトでは味わうことのできないような独特の「味」もありますし……。

大上　幸宏(22)　X68000 PROII　鹿児島県

●PDドライブは想像していたより，かなりよい機械のようです。しかし，X68000でCD-ROMを使う場合，他機種へのデータを切り出したり，吸い出したりすることのほうが多いと思います。それに，そういった人はCD-ROM→MOなどへデータを転送するでしょうから，600MバイトクラスのMOに期待というところでしょうか。まあ，よく売れているようですし，現物を１回触ってみたいですね。

奥田　直也(22)　X68000 ACE-HD, X68000 SUPER, X68030, MSX2, PC-E550　神奈川県

●「知能機械概論」を読んで，電子メールを日本中の小学校で！　というのはかなりよいと思う。現時点ではまだまだテレビ電話などは実用化が難しいわけで，大学などではすでに進んでいるインターネットを使わない手はないだろう。自分も研究室で毎日のように使い，かなりの情報を有効利用させてもらっている。国はこのような遠くの人まで手軽にコミュニケーションのとれる場をもっと理解して公開してほしいものである。

小林　佳徳(21)　X68000 XVI　新潟県

●「THE SENTINEL」についてですが，ゲーム投稿者にある程度の操作上のガイドラインを設けるべきです。たとえば，４方向の移動キーには2，4，6，8のほかにカーソルキー，M，H，K，Uなどをすべて同時にサポートしなければいけません。トリガ１は5，スペース，J，Xで，トリガ2は1，ESC，N，Zなどでしょうか。もちろん，小文字もすべて同時に対応してなければいけませんし，さらにいつでもSHIFT＋BREAKやESCなどの操作でプログラムが終了するようになっていなければいけません。こういったのはサブルーチンを１回作れば使い回せます。

ALL BASICのプログラムなら勝手に改造できるからいいのですが……。

鈴木　朝夫(21)　X68000, MZ-1500, X1 turbo Z, PC-9801RA, PC-88VA2, PC-6601SR, FM-77 AV40SX, MSXturboR, ZX-81　神奈川県

●「ビジネスショウ'95」などのショウがあると必ずレポートが掲載されます。ショウのレポート自体はいいと思うのですが，残念ながらシャープのユーザー，特にOh!Xの読者には「関係ないや！」と思われるような内容が多いように思います。こんなときは記事を掲載せずにニュースなどの項目で紹介すればよいのではないでしょうか。なにがなんでもカラーページを使うというのはどうかと思います。

壁谷　善嗣(36)　X68000 EXPERT, PC-9821As, PC-9801NS/E　宮城県

ごめんなさいのコーナー

３月号　Oh!X LIVE in '95

P.63　リスト４「FF2用コンフィグファイル」の29行目を以下のように変更してください。

```
.55c+ = ……
    ↓
.o5c+ = ……
```

５月号　Oh!X LIVE in '95

P.70　リスト６「エスプレッソ銀河用カウンタ表示」とリスト９「ミッドナイトレジスタンス用カウンタ表示」が入れ替わっていました。

７月号　ペンギン情報コーナー

P.135　セイコーエプソンのマッハジェットカラープリンタ「MJ-900C」の対応用紙サイズに間違いがありました。正しくはハガキサイズからＡ４サイズまで対応です。関係者および，読者の方々に大変ご迷惑をおかけしました。お詫びいたします。

バグに関するお問い合わせは
☎03(5642)8182(直通)
月～金曜日16：00～18：00

お問い合わせは原則として，本誌のバグ情報のみに限らせていただきます。入力法，操作法などはマニュアルをよくお読みください。

また，よくアドベンチャーゲームの解答を求めるお電話をいただきますが，本誌ではいっさいお答えできません。ご了承ください。

やっぱりアニメーションをしてみたい!?

▼いままでOh!X誌上でアニメーションを実行する方法として「DōGA CGA SYSTEM」や「AMIシステム」を紹介してきました。その「AMIシステム」を収録した1994年の3月号で福嶋氏が「今後，SCSI装置はいま以上に高速化，大容量化が進むでしょうから，AMIシステムの再生能力は，ほっといても勝手に向上するわけです」と述べています。しかし，このときから世にあるほかのマシンでは高速なSCSI装置の恩恵を受けられましたが，X68000ではそのSCSI装置に見合うだけの恩恵が受けられませんでした。

▼そんな状況下に満開製作所から高速SCSI 2ボード「Mach-2」が発売されました。8月号の新製品紹介で取り上げましたので，その速度についてはすでにご存じのことでしょう。とにかく体感でわかるくらい速度が違います。これによって，X68000にまた新たな可能性が見えてきました。今回の特集「Animation Now!」では，その可能性のひとつを実感していただけると思います。また，SEGA SATURNで多く採用されているシネパックのアルゴリズムの解析もあります。

▼7月号で行った「THE USER'S WORKS SPECIAL」は好評で，今月号ではTAKERUに登録されているソフトをいくつか紹介してみました。今回は新しいソフトはなく，少し古いソフトが主ですが，なんでも新しいものがいいとは限らないことは皆さんよくごぞんじでしょう。それに新しいソフトを制作されている方は，8月上旬までは締め切りに追われて大変でしたでしょうし。今後もできるだけ紹介していきたいと思いますので，制作者の方々の投稿をお待ちしております。

▼気象庁の長期予想では今年の夏は冷夏になるとの予想でしたが，そんな予想は見事に裏切られ昨年に続く猛暑となりました。来月号では，そんな猛暑を吹き飛ばしてくれた皆さんのカラーイラストを紹介する予定です。

▼「ハードコア3Dエクスタシー SIDE B」は著者多忙につきお休みいたします。

投稿応募要領

●原稿には，住所・氏名・年齢・職業・連絡先電話番号・機種・使用言語・必要な周辺機器・マイコン歴を明記してください。

●プログラムを投稿される方は，詳しい内容の説明，利用法，できればフローチャート，変数表，メモリマップ（マシン語の場合）に，参考文献を明記し，プログラムをセーブしたフロッピーディスクを添えてお送りください。また，掲載にあたっては，編集上の都合により加筆修正させていただくことがありますのでご了承ください。

●ハードの製作などを投稿される方は，詳しい内容の説明のほかに回路図，部品表，できれば実体配線図も添えてください。編集室で検討のうえ，製作したハードが必要な場合はご連絡いたします。

●投稿者のモラルとして，他誌との二重投稿，他機種用プログラムを単に移植したものは固くお断りいたします。

あて先

〒103 東京都中央区日本橋浜町3-42-3
ソフトバンク出版部
Oh!X「㋢㋮㋱」係

SHIFT・BREAK

▶8月号の有田先生の連載に関して面白い話題があるんだが，そのネタはあえて書かない。現在，パソコン通信では9600bpsから28800bpsが主流。研究室で155Mbpsの回線の能力を見ているが，実はたいして速くない。動画と音声をリアルタイムでやりとりするには155Mbpsでも足りないのかもしれない。（けんと）

▶夏コミであいもかわらず非常に片寄ったゲーム攻略本とオリジナル&アレンジの音楽テープを出す。ビューポの本も再版。だけど理由があって，場所もサークル名も内緒。本のほうは，ほとんどXDTPによる編集。ある意味，いまのX68000のハイエンド環境を駆使して作っています。サークルカットに私の名前があるから探してね。（瀧）

▶昔はCDは高いものだったので大切に扱ってたけど，最近部屋に雑誌の付録CDが増えてきて，扱いもわりとぞんざいになりつつある。先日，机に向かっているときにCDが落ちたので，椅子を引いて取ろうとしたら「ばりばりばり」。「あー，なんの付録CDだっけ」と思って拾ってみたら，お気に入りの音楽CDだった（涙）。すべてのCDを大切にしよう。（E.K.）

▶シネパックの320×224で秒15コマ。ソフトウェアだけならこんなもんかもしれないが，これが限界となるとちょっと寂しいものがある。ATの世界では，専用のハードなんかが最近随分安くなってきているし。やっぱりちょっと羨ましく感じてしまうな。せめてNewXにはモーションJPEGあたりのエンコーダ/デコーダを標準でつけてほしいものだ。（I.K）

▶高校時代の友人が，CMの造形の仕事をしているらしい。ドールのジュースのコマーシャルで果物の皮がむける仕掛けをつくったり，時任三郎を原始人に変身させたりしているという。高校時代から，自主制作映画などで超高校級の造形技術を見せていたSという男。そういえばあのころは僕も，意気込みだけは一人前の，熱血青年だったよなあ。（ats）

▶遂に我慢しきれずMacintoshを買った（ちなみにLC630）。すると，不思議なことにますますX68000に愛着が湧いてきてしまったのだ。通信を始めとするほとんどの環境をMacintoshに移行したにも関わらず，気がつくとX68000を立ち上げて絵やら曲やらを作っている自分がいたりする。自分の中に占めるX68000の位置を改めて確認する今日この頃。（哲）

▶GUIなんて人間の使うものではないと，COMMAND.X一筋を貫いてきたが，よりによってWindowsに触れることになった。脂汗を流し，あらゆる知り合いに深夜の質問電話を強行した結果，家のASK.SYSとキー配列互換の日本語FEPと，ED.Xコンパチ操作のテキストエディタという環境を構築することに成功した。やればできるもんだな，うんうん。（八）

▶8月号116ページの福田さん推薦のWILD TRAXは持っているが，微妙な挙動と低fpsは私には難しい組み合せ。30fpsが普通のいまこそ出してほしい。○F-1シムの歴史的傑作World Circuitの続編が完成間近。めっぽうすごいが要PC最新機種。○Rave Racerの第一印象は新鮮で悪くない。遊ぶ者に自分がうまいと錯覚させるのがこのシリーズの美点。（A.T.）

▶仕事に疲れて家に帰る。倒れこむようにして寝る。夜中に目が覚め，なにやら首のあたりが痒い。ふと手を当ててみると……，異様に腫れている。翌日が日曜日だったため，病院を探すのもひと苦労。結局はジンマシンだったのだが，特にマズイものを食べた記憶もない。せいぜい製造年月日が1年前のオレンジの缶ジュースくらいなんだけどなぁ。（高）

▶締め切りができました。もちろん「SION IV」の話です。完成のメドなどたっていませんが，人間，極限状態に追い詰められると実力以上の力を発揮するといわれています。きっとなんとかなるのでしょう（他人ごとモード）。ということで「SION IV」は，12月号の付録ディスクに収録されます（いい切っていいのか？）。（J）

▶部屋の中にはビデオが6台。うちVHSが3台で，滅多に使わないというのにすべて故障中。ビクターのビデオは電源が入らないとかフタが開かないとかシンプルなところで故障するのだが，松下のビデオは制御CPUが暴走するとかクロマ信号が反転するとか非常に複雑な症状が出る。壊れやすいというソニー製品はなぜか元気だ。（U）

▶浦和レッズのホームゲーム（電話予約）とFIFAオールスターマッチ（店頭発売）のチケット発売日が重なり，チケットぴあに並びながら新兵器のPHSでダイヤルしまくるという荒技にでた。通話状態は前日に調査ずみ。でもレッズ戦は1試合しか取れなかった。スタジアムの収容人員は2倍になったはずなのに。それにしても，3位はすごいぞ！（T）

microOdyssey

「暗号」という言葉の響きには心くすぐられるものがある。小さい頃，推理小説を読んで謎解きのひとつとして暗号が使われていた。有名なところではコナン・ドイルの「踊る人形」だろうか。実際によく使われたのは軍事目的だろう。第2次世界大戦でも使われていたし，古くはローマ帝国でも使われていたという。

ただ，古い時代から使われている暗号も，現在でも使われている暗号も根本的には同じで，暗号化する前の文章の「暗号化」と元の文章へ戻す「複号化」で成り立っている。

しかし，昔からある暗号と現代で使われている暗号には大きく異なる点がある。それは暗号鍵の存在である。というのは，暗号があまり使われない時代ならば，いくつかの暗号があればよかった。ところが，いろんなところで暗号が使われるようになると，いくつも暗号を用意するわけにはいかなくなった。そこで，ひとつの暗号方式から，暗号化と複号化の枠組みが決まるようにした。その暗号方式に与える暗号鍵(パラメータ)で異なる暗号が得られるようになったのだ。

この暗号方式の場合，暗号化や複号化のアルゴリズムは公開できるが，暗号鍵は送り手と受け手が周囲からは秘密裏に知らないといけない。「そんなことは当たり前じゃないか」と思うかもしれないが，そうではない。

最近はインターネットが話題に上ることも多く，知っている人も多いだろうが，公開鍵暗号方式と呼ばれる暗号方式がある。この方式では，暗号鍵と複号鍵が異なるのだ。従来の方式であれば，暗号鍵と複号鍵が同一で，受け手に鍵を渡す場合が問題となったが，この方式であれば，公開された暗号鍵を使って，誰でも暗号を作成して送ることができる。そして，複号化できるのは複号鍵をもっている本人だけなのだ。

公開鍵暗号方式を実際に利用したものとしてはPGP(Pretty Good Privacy)があり，フリーソフトで，ソースコードも公開されている。ただし，暗号化技術はアメリカでは輸出が制限されており，PGPも例外ではない。入手するならば，日本国内のサイトからでないと捕まることもあるそうだ。公開鍵暗号方式の詳しい話は「E-mailセキュリティ」(オーム社)などの本が参考になるだろう。

現在，金銭の決済方法として，クレジットカードがよく使われ，パソコン通信上でも一部利用されている。このときにクレジットカードの番号は保護(暗号化)されていない場合が多い。垂れ流しである。ちなみに，1993年度のVISAとマスターカードの偽造カードによる被害は4.5億～6.5億ドルに上るそうだ。この被害額はインターネットが一気に普及するならば，もっと増えていく可能性がある。

公開鍵暗号方式を使うことで，誰が送ったかを証明する認証(印鑑のようなもの)の技術も進んでいるので，クレジット番号の保護もある程度は可能である。

ただ，「日本人は安全と水はただと思っている」とよくいわれる。公開鍵暗号方式も完璧な暗号方式ではないし，いくら暗号の技術が進んだところで，結局は使う側の意識が低ければ意味がない。危険なんてそこらじゅうに転がっているのだから。　　　　(高)

1995年10月号 9月18日(月)発売

特集 Now Printing

・各種カラープリンタへの出力
・美しい出力のための減色処理と拡大処理

新連載　div-lisp入門
新刊紹介
SX-WINDOW ver.3.1開発キット
新製品紹介
ディスプレイテレビ PC-TM151

バックナンバー常備店

東京	神保町	三省堂神田本店5F	03(3233)3312
	〃	書泉ブックマートBI	03(3294)0011
	〃	書泉グランデ5F	03(3295)0011
	秋葉原	T-ZONE 7Fブックゾーン	03(3257)2660
	八重洲	八重洲ブックセンター3F	03(3281)1811
	新宿	紀伊国屋書店本店	03(3354)0131
	高田馬場	未来堂書店	03(3209)0656
	渋谷	大盛堂書店	03(3463)0511
	池袋	旭屋書店池袋店	03(3986)0311
	八王子	くまざわ書店八王子本店	0426(25)1201
神奈川	厚木	有隣堂厚木店	0462(23)4111
	平塚	文教堂四の宮店	0463(54)2880
千葉	柏	新星堂カルチェ5	0471(64)8551
	船橋	リブロ船橋店	0474(25)0111
	〃	芳林堂書店津田沼店	0474(78)3737
	千葉	多田屋千葉セントラルプラザ店	043(224)1333
埼玉	川越	黒田書店	0492(25)3138
	川口	岩渕書店	0482(52)2190
茨城	水戸	川又書店駅前店	0292(31)0102
大阪	北区	旭屋書店本店	06(313)1191
	都島区	駸々堂京橋店	06(353)2413
京都	中京区	オーム社書店	075(221)0280
愛知	名古屋	三省堂名古屋店	052(562)0077
	〃	パソコンΣ上前津店	052(251)8334
	刈谷	三洋堂書店刈谷店	0566(24)1134
長野	飯田	平安堂飯田店	0265(24)4545
北海道	室蘭	室蘭工業大学生協	0143(44)6060

定期購読のお知らせ

Oh!Xの定期購読をご希望の方は綴じ込みの振替用紙の「申込書」欄にある『新規』『継続』のいずれかに○をつけ，必要事項を明記のうえ，郵便局で購読料をお振り込みください。その際渡される半券は領収書になっていますので，大切に保管してください。なお，すでに定期購読をご利用の方には期限終了の少し前にご通知いたします。継続希望の方は，上記と同じ要領でお申し込みください。

基本的に，定期購読に関することは販売局で一括して行っています。住所変更など問題が生じた場合は，Oh!X編集部ではなくソフトバンク販売局へお問い合わせください。

海外送付ご希望の方へ

本誌の海外発送代理店，日本IPS(株)にお申し込みください。なお，購読料金は郵送方法，地域によって異なりますので，下記宛必ずお問い合わせください。

日本IPS株式会社
〒101 東京都千代田区飯田橋3-11-6
☎03(3238)0700

Oh!X 9月号
■1995年9月1日発行　定価760円(本体738円)
■発行人　橋本五郎
■編集人　稲葉俊夫
■発売元　ソフトバンク株式会社
■出版事業部　〒103 東京都中央区日本橋浜町3-42-3
Oh!X編集部　☎03(5642)8122
販売局　☎03(5642)8100 FAX 03(5641)3424
広告局　☎03(5642)8111
■印　刷　凸版印刷株式会社

満開の電子ちゃん　作・え 岡村祭

Mach-2は、残念ながらPRO、PRO2非対応となりました。MK-RGB21-15S(弐號)の正しい価格は¥5,400です。おわび申し上げます。

87号(7/18発送)には、X-BASICより4倍は速い「ぺけ-BASIC」とか、PDをサポートするドライバとか、「デスペラード誕生秘話」とか。IFM半角2書体も。

小林〈アリエヌ共和国〉猫
(東京都)

X68kのためのディスクマガジン電脳倶楽部は、読者が幅広く参加できる雑誌だ。プログラムやCG制作が苦手な人でも、読評や攻略記事などの読み物、それにPDDなど、様々な形で参加することができる。もちろん「読者」としての参加もOKさ。
マウスひとつでラクラク操作なので肉球にも優しいし、新設された「電脳掲示板」も楽しみ。
そうそう、定期購読者のみんな。アンケート葉書は忘れず出そう。それだって立派な「参加」だぜ。
さあ、あなたも購読してみませんか? あ、別冊もよろしくね。

マイコン専門ショップ

P&A

10周年記念 秋葉原店オープン!!

営業時間/AM11:00～PM7:00
(日・祭 PM6:30)
TEL 03-5294-7053
FAX 03-5294-7054
(秋葉原店は来店のみとさせていただきます。)

JR秋葉原 中央通り P&A 秋葉原店

SHARPエキスパートショップ

8/17～9/17

注目!!冬のボーナス一括払い手数料(金利)無料(平成7年9月末/10月末/11月末/12月末のいずれかをご指定下さい。)

X68030お買い得セット
(クレジット表:送料、消費税込み)

①ハードディスクセット
- CZ-500C(本体)
- 340MB(外付)ハードディスク

定価¥506,000
P&A超特価 ¥245,000
(ハードディスク540MBに変更の場合¥5,000加算して下さい。)

12回	22,200	24回	11,700	36回	8,100
48回	6,300	60回	5,300		

②モニターセット
- CZ-500C(本体)
- CZ-608D-B(モニター)

定価¥492,800
P&A超特価 ¥277,000

12回	25,100	24回	13,200	36回	9,100
48回	7,100	60回	6,000		

■②のモニター変更の場合
- CZ-615D(チューナー付)に変更の場合 ¥56,000
- CZ-621D(B)……………に変更の場合 ¥64,000

加算して下さい。

X68030/68000オリジナルセット

⊙CZ-500C	⊙CZ-500C	⊙CZ-674C	⊙CZ-674C
●HD(内蔵)500MB ●メモリー8MB増設(合計12MB) ●コプロ ●SX-WIN インストール済み	●HD(内蔵)800MB ●メモリー8M増設(合計12MB) ●コプロ ●SX-WIN インストール済み	●HD(内蔵)500MB ●メモリー6M増設(合計8MB) ●SX-WIN インストール済み	●HD(内蔵)800MB ●メモリー6M増設(合計8MB) ●SX-WIN インストール済み
特価 ¥315,000	特価 ¥338,000	特価 ¥176,000	特価 ¥199,000

⊙内蔵ハードディスク(500C、674C用)(単品)
●500MB…特価¥49,800 ●800MB…特価¥69,800
(当社取り付けの場合、¥8,000加算して下さい。)

決算大処分セール 旧シリーズ今が買いどき!!
(送料¥2,000・消費税別)(クレジット表:送料・消費税込み)

X68000 Compact XVI

- CZ-674C-H
- CZ-608D(B)

定価¥392,800
P&A特価 ¥134,000

12回	12,300	24回	6,400	36回	4,500	48回	3,500	60回	2,900

- CZ-674C-H
- CZ-608D(B)
- CZ-6FD5

定価¥492,600
P&A特価 ¥182,000

12回	16,600	24回	8,700	36回	6,000	48回	4,700	60回	3,900

決算大処分セール 旧シリーズ今が買いどき!!
(送料¥1,000・消費税別) 単品、限定

⊙PROII-HD
●CZ-663C
(ハードディスク40MB内蔵)
P&A超特価 ¥39,800

⊙PROII-HD 最強モデルセット
●CZ-663C
●メモリー11MB増設(合計12M)
●SCSIボード付
P&A超特価 ¥119,000

⊙Compact XVI
●CZ-674C
P&A超特価 ¥76,500

X68用専用ディスプレイ
- ⊙CZ-608D-B 特価¥59,800
- ⊙CZ-615D 特価¥118,000
- ⊙CZ-621D 特価¥120,000

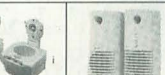
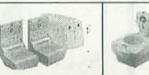
MIDIセット
- MC-6600(SNE)
- SX-68MII(システムサコム)
- MIDIケーブル

特価¥45,800

- SC-55MKII(ローランド)
- SX-68MII(システムサコム)
- MIDIケーブル

特価¥56,800
(SC-88に変更の場合¥17,000加算して下さい。)

単品
- MC-6600(SNE)…………特価¥32,800
- SC-55MKII(ローランド)・特価¥43,600
- SC-88(ローランド)………特価¥73,500
- SC-88VL(ローランド)…… 特価¥54,500

スピーカー
- SP-300(シグマ)……特価¥ 4,980
- SC-C55(AIWA)……特価¥ 5,980

ALTEC ACS300 特価¥37,000
ALTEC ACS100 特価¥16,000
YAMAHA YST-M5 特価¥6,400

X68000/68030用 メモリボード
(送料¥700・消費税別)

■I/Oデータ
- SH-5BE4-8M(30用)……特価¥39,500
- SH-6BE1-1ME(600C用)…特価¥10,200
- PIO-6BE1-AE(ACE/PRO/PROII用)・特価¥10,200
- PIO-6BE2-2ME(拡張スロット用)特価¥19,600
- PIO-6BE4-4ME(〃)特価¥33,600

■シャープ
- CZ-5BE4(30用)………特価¥39,800
- CZ-5ME4(5BE4用増設)・特価¥36,500
- CZ-6BE2A(XVI用)……特価¥38,900
- CZ-6BE2B(XVI、674C増設)特価¥37,500
- CZ-6BE2D(674C用)……特価¥20,500

モデム&FAXモデム
(送料¥1,000)

<アイワ>
- PV-BF144(ボックス型)…………特価¥17,000
- PV-AF288(推奨機種・XVI以上)…特価¥32,000

<マイクロコア>
- MC144FXe/w(ボックス型)……特価¥14,800

<オムロン>
- ME1414BII(ボックス型)………特価¥17,000
- ME2814B(推奨機種・XVI以上)…特価¥29,800
- MD-144XT10V(限定在庫限り)…特価¥30,000

MO
(送料¥1,000)

Logitec
- LMO-200(128M)……定価¥69,800▶特価¥45,800
- 340(128M)……定価¥79,800▶特価¥52,300
- 400(128/230M)定価¥118,000▶特価¥76,000
- 420(230M)……定価¥138,000▶特価¥99,000

ICM
- MO-120S-N…………定価¥74,800▶特価¥55,000
- 230S-N…………定価¥118,000▶特価¥87,000

Filo
- CS-M230PA(230MB)定価¥148,000▶¥77,800

CD-ROM
(送料¥1,000)

Logitec
- SCD-200(2倍速)……定価¥19,800▶特価¥16,800
- 420(4倍速)……定価¥34,800▶特価¥27,200
- LCD-440(4倍速)……定価¥39,800▶特価¥26,800

ICM
- CD-620S-N(4倍速)・定価¥34,800▶特価¥26,400

緑電子
- CXA-660-98(4.4倍速)定価¥39,800▶特価¥33,200
- 660-5L(〃)定価¥49,800▶特価¥44,200

東京システムリサーチ製(X SIMM)
(送料¥700・消費税別)

(X SIMM VI)
⊙X VIシリーズ専用SIMM増設式メモリボード
- X SIMM VI(634C用)…定価¥16,500➡特価¥13,000
- X SIMM VIc(674C用)・定価¥16,500➡特価¥13,000

⊙増設SIMMメモリ(72PIN)
- 4MB(70ns)……………………特価¥11,800
- 8MB(70ns)……………………特価¥27,800
- 4MB(60ns、24MHz以上用)………特価¥16,500
- 8MB(60ns、24MHz以上用)………特価¥28,000

- 6MB(60ns、メーカー純正品)……特価¥27,800

(X SIMM 10)⊙SIMM増設式メモリボード
- X SIMM 10………定価¥18,000➡特価¥15,700

⊙増設SIMMメモリ
- 1MB×2……特価¥10,000
- 4MB×2……特価¥30,000
- 10MB例 X SIMM 10+1MB×2+4MB×2‥¥55,700

X68000/68030専用ハードディスク
(送料¥1,000・消費税別)

外付

■ジェフ
- ⊙GF-340(330MB、13ms)…………特価¥25,500
- ⊙GF-540(520MB、12ms)…………特価¥28,800
- ⊙GF-730(730MB、10ms)…………特価¥39,800
- ⊙GF-1000(1060MB、9ms)…………特価¥50,800

■ロジテック
- ⊙SHD-BA340U(340MB、12ms)……特価¥26,500
- ⊙SHD-BA540U(540MB、10.5ms)…特価¥29,800
- ⊙SHD-BA1000U(1GB)……………特価¥51,800

■システムサコム(富士通純正ドライブ使用)
- ⊙HD-M520(520MB、12ms)………特価¥37,800

内蔵

■CZ-500C/300C専用
- ⊙CZ-5H08(80MB/23ms) 定価¥ 98,000▶特価¥71,800
- ⊙CZ-5H16(160MB/18ms) 定価¥135,000▶特価¥99,500

●価格は変動します。ご注文の際は必ずお電話で価格と在庫をご確認下さい。●本広告に掲載の商品には送料及び消費税は含まれておりません。

MPUアクセラレータ
（東京システムリサーチ）

⦿Xellent30（XVI用）
定価￥59,800⇨特価￥46,500

⦿Xellent30s（ACE、EXPERT（II）、SUPER用）
定価￥54,800⇨特価￥42,800

（●MPU交換に付き、保証（メーカー、当社）は付きませんので、ご承知下さい。）

P&Aならではの 新品パソコン 5年保証

《業界No.1の"P&Aメンテナンスサポート"》

最高の保証システム
① 業界最長の新品パソコン5年保証（メーカー保証1年＋P&A保証4年）（※モニター・プリンター3年間保証!! ※一部商品は除きます。）
② 中古パソコンの1年間保証（※モニター・プリンター6ヶ月間保証!!）
③ 初期不良交換OK!!（※新品商品に限らせていただきます。）
④ 永久買取保証
⑤ 配達日の指定OK!!（土曜・日曜・祭日もOK!!）
⑥ 夜間配達もOK!!（※PM6:00～PM8:00の間 ※一部地域は除きます。）

便利でお得な支払いシステム
① 翌月一括払い手数料無料（ご利用下さい。）
② 業界No.1の低金利!!
③ 月々の支払いは￥1,000より
④ 9ヶ月先からのスキップ払いOK!!
⑤ 84回までの分割、ボーナス併用OK!!
⑥ カレッジクレジット
⑦ ステップアップクレジット
⑧ ボーナスだけで10回払いOK!!
⑨ 現金一括支払いOK!!
⑩ 商品到着払いOK!!（代引き手数料が必要になります。10万円まで900円）（※商品・金額ご確認の上、銀行振込・現金書留にてご入金下さい。）

●法人向けリースシステム
業務に最適なシステムを構築します。損金処理が可能なリース契約をどうぞ。

周辺機器コーナー （送料￥1,000・消費税別）

カラーイメージスキャナ（ケーブル付）
■JX-330X（SHARP） 特価￥93,800
■GT-6500WINS（エプソン） 特価￥59,800

カラーイメージジェット 限定5台

■IO-735X-B 定価￥248,000 特価￥89,000

ビデオスキャナー
■CZ-6VS1 定価￥178,000 特価￥129,000

FDD（5インチ×2基）
■CZ-6FD5 定価￥99,800 P&A超特価 ￥49,800

プリンター（ケーブル付）
- MJ-700V2C（エプソン）…特価￥53,300
- MJ-800C（エプソン）…特価￥61,300
- MJ-500C（エプソン）…特価￥40,300
- MJ-900C（エプソン）…特価￥81,300
- MJ-5000C（エプソン）特価￥139,800
- BJC-400J（キャノン）…特価￥41,300
- BJC-600J（キャノン）…特価￥53,300
- BJC-35V（キャノン）…特価￥44,300
- BJ-30V（キャノン）…特価￥33,300

ペン&タブレット

■Drawing Slate（NS・カルコンプ）
●31090SER（6×9） 定価￥74,800 ▶特価￥48,800

- CZ-6BV1………定価￥21,000▶特価￥15,900
- CZ-8NM3………定価￥ 9,800▶特価￥ 7,200
- SH-6BF1………定価￥49,800▶特価￥36,500
- CZ-6BS1………定価￥29,800▶特価￥21,500
- CZ-8NJ2（限定）…定価￥23,800▶特価￥13,800
- CZ-6CS1（674C用）・定価￥12,000▶特価￥ 8,900
- CZ-6CR1（RGBケーブル）・定価￥ 4,500▶特価￥ 3,600
- CZ6CT1（テレビコントロール）・定価￥ 5,500▶特価￥ 4,400
- CZ-5MP1（X68030用）・定価￥54,800▶特価￥42,000
- TN-800TVEM（ビデオスキャンコンバータ・東京ニーズ）……………特価￥27,800

送料￥700・消費税別

■システムサコムボード
- SX-68MII（MIDI） 定価￥19,800 特価￥13,500
- SX-68SC（SCSI） 定価￥26,800 特価￥17,500

X68000用ソフトコーナー （送料￥700・消費税別）

〈シャープ〉
- MUSIC PRO68K（MIDI）（CZ-247MSD）……特価￥20,500
- CANVAS PRO68K（CZ-249GSD）特価￥22,000
- Easypaint SX-68K（CZ-263GWD）……特価￥ 9,800
- Easy draw SX-68K（CZ-264GWD）・特価￥15,300
- New Print Shop Ver.2.0（CZ-265HSD）……特価￥15,400
- Press Conductor PRO68K（CZ-266BSD）……特価￥22,000
- CHART PRO68K（CZ-267BSD）・特価￥29,800
- EG-Word（CZ-271BWD）……特価￥44,900
- Communication SX68K（CZ-272CWD）……特価￥14,500
- Datacalc SX-68K（CZ-273BWD）……特価￥44,000
- MUSIC SX68K（CZ-274MWD）・特価￥29,300
- SOUND SX68K（CZ-275MWD）・特価￥11,500
- フォント・アンド・ロゴデザインツール SX-68K（CZ-282BWD）……特価￥22,000
- BUSINESS PRO68K（CZ-286BSD）……特価￥20,500
- SX-WINDOWディスクアクセサリー集（CZ-290TWD）……特価￥11,500
- XDTP-SX68K（CZ-291BWD）……特価￥26,900
- C-Compiler PRO68K Ver.2.1（CZ-295LSD） NEW KIT……特価￥32,500
- SX-WINDOWS Ver.3.1（CZ-296SS/SSC）……特価￥17,600

〈計測技研〉
- Double Bookin……特価￥ 9,600
- CD-ROM Driver V.2.0……特価￥ 3,800
- シャーペンワープロパック……特価￥ 5,400

〈その他〉
- F-Card V5 for X68K（クレスト）……特価￥ 9,600
- F-Calc for X68K（クレスト）……特価￥11,000
- たーみのる2（SPS）……特価￥13,000
- MU-1GS（サンワード）……特価￥21,000
- マチエール V2.1（サンワード）……特価￥28,800
- Z's STAFF PRO68K Ver.3.0（ツァイト）……特価￥37,500
- Z's TRIPHONYデジタルクラフト（ツァイト）……特価￥27,000
- XL/Image（IMAGICAテクノシステム）……特価￥46,000

〈ゲーム〉在庫限り
- 魔法大作戦（X68/5"）……特価￥ 7,300
- パックランド（X68/5"）……特価￥ 6,200
- 餓狼伝説（X68/5"）……特価￥ 6,600
- スーパーストリートファイターII（X68.5"）……特価￥ 7,300

P&A 株式会社ピー・アンド・エー
〒124 東京都葛飾区新小岩2丁目2番地20号
●営業時間：AM10：00～PM7：00 日・祭：AM10：00～PM6：00
☎03-3651-0148（代）
●定休日/毎週水曜日 FAX.03-3651-0141 MAC/DOS Vフロア☎03-3655-4454

全国通販 ★頭金なし！★即日発送

●お近くの方はお立寄り下さい。専門係員が説明いたします。
●本体単品で特価で受付します。詳しくは電話にてお問合せ下さい。
●ビジネスソフト定価の20%引きOK！TELください。

※お支払いは、便利な商品到着払い〈手数料（10万円まで900円）要〉をご利用下さい。

P&A特選 今月の中古特選品

- 単品 ●CZ-500CB ￥165,000
- ●CZ-623C ●68000専用モニター付 ￥89,000
- ●CZ-653C ●68000専用モニター付 ￥69,000

新品限定
- ●CZ-652C……￥46,800
- ●CZ-653C……￥47,800
- ●CZ-663C……￥49,800

- CZ-600C‥￥40,000
- CZ-601C‥￥40,000
- CZ-611C‥￥45,000
- CZ-652C‥￥39,800
- CZ-612C‥￥60,000
- CZ-603C‥￥53,000
- CZ-653C‥￥41,000
- CZ-612C‥￥65,000
- CZ-623C‥￥75,000
- CZ-674C‥￥59,800
- CZ-634C‥￥110,000
- CZ-644C‥￥145,000

※上記は単品価格、モニター別売。

高額買取り（新品もOK）格安販売

■まずはお電話下さい。下取り専用 買取り電話▶ ☎03-3651-1884 FAX.03-3651-0141

買取り価格…完動品・箱／マニュアル／付属品の価格です。中古販売…1年間保証付。

●下取りの場合…価格は常に変動していますので査定額を電話で確認してください。（差額は、P&A超低金利クレジットをご利用ください。）
●買取りの場合…現品が着き次第、3日以内に高価買取金額を連絡し、振込み、又は書留でお送り致します。

●最新の在庫情報・価格はお電話にてお問い合せください。
●買い取りのみ、または、中古品どうしの交換も致します。詳しくは電話にて、お問い合せください。
●価格は変動する場合もございますので、ご注文の際には必ず在庫をご確認ください。
●本商品の掲載の価格については、消費税は、含まれておりません。
●現金書留及び銀行振込でお申し込みの方は、上記商品の料金に3%加算の上でお申し込み下さい。詳しくは、お電話でお問い合せください。

P&Aオリジナル特選パソコンラック&OAチェアー （消費税込み）（送料無料、離島を除く）

①￥17,304（スライド式・キーボートテーブル&マウステーブル）
※キャスター付、5段、17"モニターOK、色（グレー）。※上から2番目棚板移動可能。

②￥12,360（マウステーブルスライドOK）
※キャスター付、4段、17"モニターOK、色（グレー）。※スライドマウステーブル、中棚板は2段階移動可能。

①￥4,944 ●布張り色（グレー）●ガス圧シリンダー

②￥6,283 ●肘付 ●布張り色（グレー）●ガス圧シリンダー

※ラック、チェアー持ち帰り可能です。ご来店下さい。

通信販売お申し込みのご案内

〔現金一括でお申し込みの方〕
●商品名およびお客様の住所・氏名・電話番号をご記入の上、代金を当社まで現金書留でお送りください。（プリンター・フロッピーの場合、本体使用機種名を明記のこと）

〔クレジットでお申し込みの方〕
●電話にてお申し込みください。クレジット申し込み用紙をお送りいたしますので、ご記入の上、当社までお送りください。●現金特別価格でクレジットが利用できます。残金のみに金利がかかります。●1回～84回払いまで出来ます。但し、1回のお支払い額は￥1,000円以上。

〔銀行振込でお申し込みの方〕
●銀行振込ご希望の方は必ずお振込みの前にお電話にてお客様のご住所・お名前・商品名等をお知らせください。（電信扱いでお振込み下さい。）

〔振込先〕さくら銀行 新小岩支店
当座預金 2408626 （株）ピー・アンド・エー

超低金利クレジット率

回数	3	6	10	12	15	24	36	48	60	72
手数料	2.6	3.0	4.2	4.89	6.5	10.0	14.3	18.9	24.3	31.8

（※車でお越しの場合は北海道拓殖BK前の新小岩駐車場をご利用下さい。）

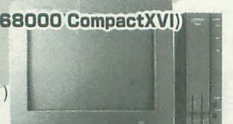
ツクモの68フロア(本店4F)は周辺ソフトもスゴイ!!

真・ツクモ・ザ・バーゲン開催中!

TSUKUMO TSUKUMO TSUKUMO TSUKUMO TSUKUMO TSUKUMO TSUKUMO

ツクモイメージキャラクター 生田智子

お申し込みは今すぐ！ 受注専門フリーダイヤル **0120-377-999**

X680x0シリーズ

X68030 お勧めのセット2

CZ-500C-B・・・・・・ ¥398,000

外付500MB ハードディスクサービス

特価 ¥228,000 ※モニタ別売です

満開製作所の商品も取扱中！

X68000 CompactXVI 24MHz改

RED ZONE・・・・・・・・・・・・・・ 特価 ¥ 98,000

RED ZONE(2DD)・・・・・・・・ 特価 ¥ 103,000

満開製外付け5インチFDD

MK-FD1 ・・・・・・・・・・・・・・・・・・・ 特価 ¥ 39,800

X680x0シリーズ用RAMボード

SH-6BE1-1ME ・(CZ-600C専用) ・・・・・・・・・・・・ ¥10,500
PIO-6BE1-AE ・・・(ACE/PRO/PRO2シリーズ用) ・・ ¥10,500
PIO-6BE2-2ME (拡張スロット用) ・・・・・・・・・・ ¥19,800
PIO-6BE4-4ME (拡張スロット用) ・・・・・・・・・・ ¥33,800
SH-5BE4-8M (X68030シリーズ用) ・・・・・・・・・・ ¥42,800
X SIMM VI ・・・・・・・(XVI専用) ・・・・・・・・・・・・・・ ¥13,200
X SIMM VIc・・・・・・(CompactXVI専用) ・・・・・・・・ ¥13,200
X SIMM10-8MBSIMM(拡張スロット用8MB) ・・・ ¥51,800
X SIMM10+10MBSIMM(拡張スロット用10MB) ¥62,800

★当社でお取り扱いの商品は、お客様による改造後での動作保証は、一切致しません。

XsimmVI/VIc/TS-6BS1mkII用

8MB72Pin60nsパリティ無しSIMM

特価 ¥35,000

★各SIMMマザーカードとセットの場合

特価 ¥33,000

MPUアクセラレーターカード

XVIユーザー様に続いてACE/EXPERT/SUPERユーザー様へ朗報！
MC68030環境＋αがお手ごろ価格で新登場です！

MC68000モードとMC68030モードをソフトウェアにて切り替え可能ですので、既にお手持ちのソフトが動作しなくなる心配はありません。取付はドライバー1本でOKです。通常の動作速度向上はもちろん！レンダリング等の高精度演算処理に威力を発揮するMC68030モード用コプロセッサを登載しておりMPUからダイレクトに制御する専用プログラムがあれば、さらに動作速度が向上します。

売れてます!! CZ-601/611/602/612/603/613/604/623専用
T.S.R製 **Xellent30s** 定価¥54,800 特価 ¥41,800

大好評発売中 CZ-634/644専用
T.S.R製 **Xellent30** 定価¥59,800 特価 ¥45,800

取付費別（店舗持ち込み時¥5,000、7日程度の日数を頂きます）※Human Ver3.0以外のOSは1995／7／12現在対応しておりません。

DSPプロセッサカード

可能性は無限大!!DSPを操り高速演算、EIAJ光デジタル入力で高品質音声録音ができる! また、別売り赤外線I/Fで、リモコン制御、電子手帳データー交換……なども。

GRAVIS製 **AWESOME-X** 定価¥89,800 特価 ¥79,800

マウス延長ケーブル(1.5m) **TS-MEXCB**・・・・特価 ¥1,880

X68000Compact/RED ZONE用 内蔵6MB+FPUボード

TS-6BE6DP

※FPUにMC68882を使用しているため、HumanVer3.0より前に付属していたFLOAT3.Xでは使用できませんのでご注意ください。
★大好評につき、若干納期を頂く場合がございます。ご了承下さい。
定価¥64,800

特価 ¥57,800

キーボード延長ケーブル(1.5m)

TS-KEXCB 特価 ¥1,880

X680x0ユーザーの為の ツクモオリジナルシリーズ

ジョイスティックパラレルインターフェイス

●拡張スロットを使用しません。ジョイスティック端子に接続できるパラレルインターフェイスです。これでスキャナも高速で取り込みが可能になります。★取り込みソフトェア及びサンプルソース付属。

TS-JPIFE (EPSONスキャナ対応用) 定価¥17,800 特価 ¥14,800

Matier Ver.2.1対応！

TS-JPIFS (CZ-8NS1対応用) 定価¥17,800 特価 ¥14,800

ツクモオリジナルX680x0 HG

		本体	HDD	RAM	コプロ	特価
X68030	HG500	CZ-500	515MB	12MB	○	¥299,000
	HG320	CZ-500	324MB	12MB	○	¥280,000
X68000	HG500	CZ-674	515MB	8MB	×	¥188,000
	HG320	CZ-674	324MB	8MB	×	¥168,000

★HGシリーズのお問い合わせはニューセンター店（担当 伊藤）まで

モデム

US Robotics **Sportster 28800FAX** 特価 ¥34,800

US Robotics **COURIER V.34 TERBO** 特価 ¥53,800

AIWA **PV-BF144** 特価 ¥15,800

OMRON **ME1414BII** 特価 ¥15,800

ターミナルソフト（モデムを使用する為に、必要な通信ソフト）

SPS **た～みのる2** ・・・・・・・・特価 ¥13,000

SHARP **Communication SX-68K** ＊要SX-Ver.3.0以降 ・・・・・・・・・・・・・・・特価 ¥15,800

プリンター

表記のないものはカラー、ケーブル別売 セット特価¥3,000!! ただしREDZONE用は¥5,500

●EPSON

MJ-500C ・・・・・・・・・・・・・・特価 ¥39,800
MJ-800C ・・・・・・・・・・・・・・特価 ¥63,800
MJ-900C ・・・・・・・・・・・・・・特価 ¥84,800

●Canon

BJC-35v ・・・・・・・・・・・・・・特価 ¥43,800
BJC-400J ・・・・・・・・・・・・・・特価 ¥39,800
BJC-600J ・・・・・・・・・・・・・・特価 ¥52,800
BJ-10vLite (モノクロ) ・・・・・・・・・・・・・・特価 ¥23,800

スキャナ

SHARP **JX-330X** 台限 特価 ¥89,800 ●SCSI接続 対ハーフケーブル付
SHARP **CZ-8NS1** 特価 ¥44,800 ●接続ケーブル付
EPSON **GT-6500WINS** 特価 ¥49,800 ●SCSI接続 ケーブル別

HDD ★新製品続々登場中!お問い合わせ下さい!

I/Oデータ **HDS-540M**(H-Hケーブル付) ・・・・・・ (540MB) 特価 ¥26,800
I/Oデータ **HDS-1G**(H-Hケーブル付) ・・・・・・ (1GB) 特価 ¥55,800
KONIC **VIP-340CX**(H-Hケーブル付) ・・・・ 台数限定 (340MB) 特価 ¥25,800
KONIC **VIP-540CX**(H-Hケーブル付) ・・・・ (540MB) 特価 ¥31,800
KONIC **VIP-1080CX**(H-Hケーブル付) ・・・・ (1GB) 特価 ¥55,800

★VIPシリーズはブラックなので68にぴったり！

ディスプレイ

CZ-608D (14型カラーディスプレイ) ・・・・・・・・・・特価 ¥66,000
CZ-615D (15型カラーディスプレイテレビ) ・・・・・・・・・・特価 ¥132,000
CZ-621D (21型カラーディスプレイ) ・・・・・・・・・・特価 ¥125,000

CD-ROM ★新製品続々登場中!お問い合わせ下さい!

Logitec(ケーブル別売) (4.4倍速) 特価 **LCD-440** ・・・・ ¥23,800
メルコ(H-Hケーブル付) (4倍速) 特価 **CDS-4E** ・・・・・・・・ ¥24,800
緑電子(H-Hケーブル付) (4.4倍速) 特価 **CXA-660** ・・・・ ¥32,800
Panasonic(F-Hケーブル付) (4倍速) 特価 **LK-RC504NZ** ・・ ¥28,800
I/Oデータ(H-Hケーブル付) (4倍速) 特価 **CDG-TX4** ・・・・ ¥27,800
緑電子(H-Hケーブル付) (6倍速) 特価 **CXA-900** ・・・・ ¥47,800

CD-ROMドライバー別売,セット特価¥4,000!!

MO ★新製品続々登場中!お問い合わせ下さい!

Logitec(ケーブル・メディア別) **LMO-400** ・・・・・・・・・・ (230MB) 特価 ¥59,800
Logitec(ケーブル・メディア別) **LMO-450H** ・・・・・・・・・・ (230MB) 特価 ¥74,800
ELECOM(H-Hケーブル・メディア付) **EMO-2300S** ・・・・・・・・・・ (230MB) 特価 ¥79,800

7月12日現在、メーカー未出荷のものに関しては、出荷時の仕様変更によりお手持ちの機種で動作しない場合もあります。お問い合わせ下さい。

【東 京】●パソコン本店（各種パソコン・周辺機器）●本店IIWindowsタワー（パソコン・ワープロ）●DOS／Vパソコン本館（DOS／Vパソコン・Mac･下取り）●万世店（総合通信機器）●5号店（ビデオ・ムービー・CS）●ソフト8号店（ゲーム機・ゲーム用ソフト）●買取センター（ゲーム機・ゲーム機用ソフト買取り）●ニューセンター店(各種パソコン･中古･下取り･買取り) 【名古屋】●名古屋1号店(パソコン全般)●名古屋2号店（パソコン全般･総合通信機器・ビデオ） 【札 幌】●札幌店（パソコン全般･総合通信機器）●DEPOツクモ札幌（パソコン全般）

ツクモの先取り！新製品情報 〜これは待ち切れない!!

どしどしお問い合わせ下さい!!

KUMO TSUKUMO TSUKUMO TSUKUMO TSUKUMO TSUKUMO TSUKUM

受付時間 (平日) AM10:45〜PM7:30 (日・祝) AM10:00〜PM7:00 通販の(休)第3木曜日

『FAX24時間お見積もり受付』 03-3255-4199 お名前.住所.電話番号.FAX番号をご記入の上ご依頼下さい。

ツクモグローバルJCBカード

JCBならではの国内・海外サービスにツクモオリジナルの特典をプラス。ツクモ各店にある入会申込書にてお申し込み下さい。くわしくはグローバル事務局03(3251)9898又は各店へ。
※ジャックス・VISA・セントラル・マスターも取り扱っております。

ツクモオリジナル

これぞ、スロット節約の定番！

SCSI+SIMMマザーカード

TS-6BS1mkII'

★X68000PROシリーズにも対応
★SCSIROMをさらにチューンアップ

予価 ¥39,800 9月中旬発売予定

楽しみは後に…?! (もう少しお待ちください!!)

TS-6BS1 mkII用チューンROM

予価 ¥9,800

9月中旬発売予定 楽しみは後に…?! (もう少しお待ちください!!)

X68030シリーズの最大メモリがついに12MBを超える！

16MB増設メモリ

TS-6BE16(Xe30) Xellent30用
TS-6BE16(CZ50) CZ-500C/510C用

予価 ¥89,800

9月中旬発売予定 楽しみは後に…?! (もう少しお待ちください!!)

※フラットイメージでの増設ではありません。このメモリボードへのアクセスには対応ソフトが必要です。

●満開製作所●

内蔵SCSIの転送速度では満足出来ない方へ！

SCSI2(FASTSCSI)カード

Mach-2 ★X68030/REDZONE/XVIユーザーにおすすめ！

特価 ¥28,800

●NS Calcomp●

さらにお手軽になった、お絵描きアイテム!!

コードレスタブレット

DrawingSlateII 定価¥34,800

特価 ¥31,800

ソフトウェア 特価

品名	特価	品名	特価	品名	特価
SX-WINDOW Ver3.1システムキット	¥18,200	SX広辞苑(CD-ROM別)	¥17,800	DataCalc SX-68K	¥47,800
SX-WINDOWデスクアクセサリ集	¥11,800	シャーペンワープロパック(要SXver.3.1)	¥6,120	フォント&ロゴデザインツール書家万流SX-68k	¥23,800
Easydraw SX-68K	¥15,800	EGWord SX-68K	¥47,800	C COMPILER Ver2.1 NEWKIT	¥35,800
Easypaint SX-68K	¥10,200	SX-WINDOW開発キット	¥31,800	Matier Ver2.1	¥29,800
Communication SX-68K	¥15,800	MUSIC SX-68K	¥30,400	XL/Image	¥49,300
		XDTP SX-68K	¥28,000	CD-ROM Driver	¥4,320

MIDIコンピュータ特選セット(これは大特価!!)

SC-55mkII セット
SC-55mkII ¥55,000
SX-68MII ¥19,800
特価 ¥57,800

SC-55mkII 互換セット
MC6600 ¥49,800
SX-68MII ¥19,800
専用MIDIケーブル ¥2,200
特価 ¥46,800

SC-88 セット
SC-88VL ¥69,000
SX-68MII ¥19,800
MIDIケーブル(1m) ¥1,000
特価 ¥69,800

映像関連機器

動画を始めてみませんか？

ビデオ入力ユニット**CZ-6VS1** 定価¥178,000
MC68EC020(25MHz)の32BitMPUを搭載し、SCSIを介してパソコンへデータを転送。動画・静止画を簡単に保存出来るアプリケーションソフト「ライブスキャン」を標準装備。1,677万色まで対応し、最大640×480ドットの高解像度で、高速取り込が可能です。但しX680x0シリーズでご使用の場合には6万5千色までの表示となります。

特価 ¥135,000

多機能対応型スキャンコンバーター

電波新聞社 **XVGA-1V** 定価¥66,800

特価 ¥56,700

X680x0シリーズやその他のパソコンの水平周波数(24KHz/31KHz)をNTSC標準信号に変換するスキャンコンバータユニットですので、家庭用テレビやビデオ・デッキで映像を表示または録画することができます。また、ビデオプリンターを使えば画面のハードコピーも可能です。

XVGA OVERLAY UNIT 定価¥45,800
「XVGA-1V」に接続して、パソコンとビデオの映像を合成する拡張機器です。
X68でコントロールできる！(RS-232c接続)

特価 ¥38,900

XAV-2s

定価¥10,000
パソコンやゲーム機から出力される水平同期周波数15kHzのアナログRGBの映像信号を家庭用テレビで見られる映像信号(ビデオ/Sビデオ)に変換する装置です。

特価 ¥8,500

秋葉原

至お茶の水 昌平橋通り パソコン本店 5号店 本店IIWindowsタワー 牛丼のトントン ツクモソフト8号店 買取センター 万世橋警察 万世店 三菱銀行 中央通り DOS/Vパソコン本館 至神田 秋葉原駅 JR山手・京浜東北線 ニューセンター店 至浅草橋

名古屋

名古屋2号店 名古屋1号店 若宮大通り 矢場町

札幌

ツクモ札幌店 DEPO ツクモ2番街店

お支払い方法

あなたのご都合に合わせていろいろ選べます。

クレジット払い

月々¥3,000以上の均等払いも頭金なし。夏・冬ボーナス2回払いもOK！

カード払い

¥5,000以上
通信販売での御利用カード
ツクモグローバルカード・セントラル・ジャックス
※御本人様より電話で通信販売部へお申し込み下さい。

各種リース払い

詳しくは各店にご相談下さい。

現金書留払い

〒101-91 東京都千代田区神田郵便局私書箱135号
ツクモ通販センター Oh!X係

代金引き換え配達

お申し込みは電話1本でOK!
配達日の指定もできます。

銀行振込払い

事前にTELでお届け先をご連絡下さい。
三和銀行 秋葉原支店
(普) 1009939 ツクモデンキ
※振込手数料はお客様の負担となりますすのでご了承下さい。

商品についてのお問い合わせは各店に

秋葉原

(営)平日AM10:45〜PM7:30・祝AM10:00〜PM7:00
(休)第3木曜日

ツクモパソコン本店 4F
03-3253-1899
03-3253-5599(代)

ツクモニューセンター店
03-3251-0987

名古屋

(営)平日AM10：30〜PM7：30 土・日・祝AM10:00〜PM7:30

ツクモ名古屋1号店
052-263-1655
第1アメ横ビル内 (休)8月は(火)・21・23日 9月は(火)

ツクモ名古屋2号店
052-251-3399
第2アメ横ビル内 (休)8月は(水)・24・25日

札幌

(営)平日AM10：45〜PM7：30日・祝AM10：15〜PM7：00

●両店ともX68関連商品はお取り寄せのみ(展示等はありません)となります。ご了承下さい。

ツクモ札幌店
011-241-2299(休)木曜日

DEPOツクモ2番街店
011-242-3199(休)木曜日

★商品はお電話受け付けより、標準日数3日〜1週間でお届け致します。(一部地域を除く)
★表示価格には消費税は含まれておりません。

安いのに親切
TSUKUMO

九十九電機株式会社

満足度120％アップ（当社比）
ますます充実、ソフトバンクのゲーム本！

GAME BEST SELECTION

超話題の純国産シミュレーションソフトを完全攻略!!

Tower［タワー］公式パーフェクトガイド

昨年発売された中で最も優れたソフトに与えられる権威ある「Codies賞」を受賞した、大ヒット純国産シミュレーションゲーム「Tower」公式完全ガイド。最高グレードである〈Tower〉の称号をもらうまでの様々なテクニック、自分の好きなビルを建築するためのノウハウなど、「Tower」のすべてを徹底解説！

山猫有限会社 著
A5判・定価1,600円

© OPeNBooK

キミだけの遊園地を作ろう！

theme PARK［テーマパーク］パーフェクトガイド

誰にでも簡単に遊べて、それでいて奥が深い。それがブルフロッグのシミュレーションゲーム「themePARK」です。本書はこのthemePARKの攻略法を、コミックやイラストなどをふんだんに用いて、わかりやすく解説します。この一冊で、キミも遊園地王を目指せ！

山猫有限会社 著
A5判・定価1,600円

© 1994,1995 Bullfrog Productions,Ltd. © 1995 Electronic Arts.

好評発売中

蓬萊学園108の謎
柳川房彦 監修
ゆうせぶん/賀東招二 著
定価1,500円

「ペンドラゴン」リプレイ 三つの槍の探索
健部伸明 監修
佐藤俊之 著
定価1,800円

「ファー・ローズ・トゥ・ロード」リプレイ RPGセッションガイド
遊演体 監修
司史生/ゆうせぶん 著
定価1,600円

SIMCITY 2000 パーフェクトガイド
中島理彦 著
定価1,600円

あの「Hourai Times」を再現！
蓬萊学園DX1

遊演体 編著
柳川房彦 監修
A5判・定価1,300円

テーブルトークRPG蓬萊学園の世界を解説。架空の高校「蓬萊学園」の内部で発行されている月刊誌「Hourai Times」を再現。特集、広告、対談、人生相談、読者ページなどの、雑誌記事らしい企画を中心に作成。また、「蓬萊学園の冒険!!改訂版」のリプレイ、Q&A等のルールフォロー企画と、コミックや短編小説等も盛り込み、蓬萊ワールドの楽しみ方を紹介。

近刊

ウィザードリィ大事典

9月中旬発売予定

ヘッドルーム 編著
B5判・予価3,500円

Wizardryシリーズ全作品の解法が、これ一冊で全てわかる。モンスター、アイテム、種族、魔法などのデータと、全MAP攻略を中心に、各シナリオのストーリー紹介や裏技なども盛り込んだ、完全攻略本。

Jリーグサッカー プライムゴール3 スーパーガイド

8月上旬発売予定

定価880円

ヨッシーアイランド スーパーガイド

9月中旬発売予定

予価880円

■定価は税込みです　■お近くの書店でお求めください

ソフトバンク株式会社／出版事業部 SOFT BANK
販売局 TEL.03-5642-8101

夏の海、板はモザイク、ホットジャワ、ジャストのX68kペリフェラル

さて、今頃は夏真っ盛り、気分はなつなつって感じの日々が続いている（と思慮される）訳ですが、この号が発売される頃は友人総出で晴海界隈といった方も数多くいらっしゃれるかと存じます。アウトドア慣れしていない方も多数いらっしゃると思います。その筋の皆様の健康を願ってやみません。さて担当の方ですが、先日某神奈川県警の交通安全センターに行って行政処分に付されてまいりました。累計７点、免許停止30日間。もちろん道交法103条８項の規定に基づく講習を受講して短縮29日の決定を受けてまいりました。この講習費用6,000円で、吉野屋の牛丼大盛＋お新香10食分だったなあ、と貧乏人特有の金勘定をしつつ、先月時点より貯金は５点とさらに増え（笑）、次の60日免停も晴れて確定（しくしく）。招待状はいつかとビクビクする今日この頃でございます。くー、杉◯署も◯調署も武◯野署もみんな嫌い…じゃなくて大好きですよ〜、今後も首都の交通安全のため頑張って下さいね。…で、反則金まだ払ってなかったっけ。

★X680x0を楽しむすべての人々に。★X68000、初代機は120nsのDRAMを搭載していました。モデルを追って実装されるDRAMの速度はより高速なものへと変っていきましたが、基本的なアーキテクチャーは大きな変化をしていません。正しい互換性の思想です。★しかし、世は流れて行きます。メモリーの高速化に代表されるが如く、設計ルールの微細化がもたらすデバイスのクロックスピードの向上は著しいものがあります。0.5μmを切るルールが当たり前の様に採用され、家庭用ゲーム機だって、64bitRISC＋20MHzオーバーのクロックスピードを持つ、そんな時代になりました。★そしてクロックスピード10MHzのX68000、今さらながら速い処理速度とは言えなくなりました。そして、X68000のスタンダードアーキテクチャーでもある訳です。★X68000の高速化として、一方でメーカーとしてのXVIやX68030といった上位機種へのアプローチがなされてきました。その一方で在来機は取り残されて行くわけですが、その絶対数の少なさと美しいアーキテクチャーによって、PC/AT互換機のような大きな淘汰の波もなく、「コンピューティングを楽しむ」純粋なホビーマシン、そして一部のFA用コンピューターとしての地位を未だに維持しています。★このX68000をもっと楽しくする、あるいは楽しめるマシンにしたい、そういった考えがこれらのペリフェラルを造り上げるきっかけでした。初心者からベテランまで、アプリケーションプレイヤーからプログラマー、エンジニアまで、もっと楽しく、もっと幸せになれるよう努力を重ね、そして今後も重ねていきたいと思います。★バスタイミングを維持したまま68000の倍速化を図り、かつ、ハードウェアに手をかけることに不慣れな方にも、手軽にMPUのクロックアップを図っていただけることを目的としたH.A.R.P、そしてそのアドバンテージを最大限に発揮すべく、メモリーアクセスのクロックサイクルの短縮化、そして72ピンSIMMを実装できる拡張メモリーカードER10、初代機でも、最新機種でも、変らず楽しめる、幸せになれる、そういった製品を造って行きたいと思います。★60日免停が近ずくと、人間、こうも神妙になるものなんでしょうか？。やっぱり初心が大切だってことでしょう（笑）。今後とも変らないご晶屓を。

MPUアクセラレーター H.A.R.P for MC68000
型番：DCMA00D1 定価￥29,800　対応機種：X68000初代,ACE,EXPERT,SUPER

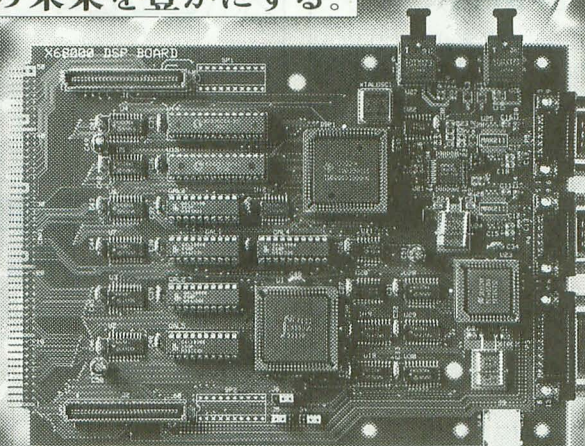
拡張SIMMメモリーボード ER10S
型番ER10S0n（SIMM未実装）定価￥14,800；ER10SDn（4MByte SIMM１枚実装済）定価￥39,800　対応機種：X680x0全機種（定価はすべて税別）

★そして、最新機種でもっと幸せになりたい方へ。

拡張I/Oスロット ESX68
型番：ESX68L4　予価￥39,800　対応機種：X680x0全機種

MPUアクセラレーター H.A.R.P-FX（H.A.R.P for MC68030）
型番：DCMA30F1　予価￥54,000
対応機種：X68030をはじめ、MC68030（PGAソケット）が採用されたコンピューターシステム（供給クロック25MHz以下）

サポート　(有)エヌ・エム・アイ
開発・販売　(株)ジャスト　JAST
〒156　東京都世田谷区宮坂3-10-7 YMTビル3F
Phone.03-3706-9766　FAX.03-3706-9761　BBS.03-3706-7134

ＤＳＰがＸ６８０ｘ０の未来を豊かにする。

Ｘ６８０ｘ０を進化させる高速演算ＤＳＰプロセッサーボード「ＡＷＥＳＯＭＥ－Ｘ」登場。

この一枚のボードが、Ｘ680x0の未来を拓く。高速演算処理によるＣＧのクォリティアップや制作時間の短縮、128,000bpsのＲＳ-232Ｃ高速通信、48ｋHz高音質デジタルサンプリング、赤外線通信機能などに対応した多機能・高性能化を実現。ＤＳＰ(Digital Signal Processor)搭載の高速演算プロセッサーボード「ＡＷＥＳＯＭＥ-Ｘ」が、あなたのＸ680x0を、新たな可能性の世界へと進化させます。

■主な仕様 ●DSP:TEXAS INSTRUMENTS社 TMS320C26B-40MHz ●RAM:DSPワーク64KB, I/F 4KB●RS-232C:D-sub9pin×2●EXT 1:EIAJ準拠 光デジタルオーディオI/F入出力端子●EXT 2:赤外線通信用I/F●EXT 3:拡張I/F ■付属ソフトウェア(予定) ●FLOAT2.X互換FLOATドライバ●DSP直接制御FLOATドライバ●高速シリアルドライバ●シリアルMIDIドライバ●PCMドライバ●JPEGデコーダ／エンコーダ●セルフプログラムチェック●ベンチマークプログラム■オプション(予定) ●MIDIドーターボード(純正MIDIボード互換)●赤外線通信ユニット(赤外線通信、電子手帳とのリンク)●Maximum Over Drive Processorボード(TMS320C3x搭載アクセララレータボード)
標準価格￥８９，８００（税別）

DSP INJECTION forX680x0
AWESOME-X
X680x0 用ＤＳＰ高速演算プロセッサーボード

GRAVIS
企画・開発/(有) グラビス〒213 神奈川県川崎市高津区坂戸3-2-1 かながわサイエンスパーク東棟513　tel:044(812)7499 FAX:044(813)7243

9月11日発売

Version Up!!

SXパワーアップ委員会 シャーペンワープロパックVer.2.0

予価￥9,800

いよいよ、SXパワーアップ委員会シリーズの第2弾、「シャーペンワープロパックVer.2.0」の登場です。シャーペンを限りなくワープロに近づける「シャーペンワープロパック」が、Ver.2.0になってさらに高機能に！Ver.2.0の目玉は、ずばりこの4つ。

☆最新カラープリンタ(MJシリーズ、BJCシリーズ)を充実サポート！
☆レイアウトモードで印刷時の状態を確認しながら編集可能！
☆マスターフォームの導入でDTPライクなページ作成！
☆CD-ROM辞書検索機能を追加！

では、今月は拡張されたワープロ機能について(^_^)

●レイアウトモードの追加
印刷時のイメージそのままで編集する「レイアウトモード」を追加。

●マスターフォームのサポート
ページのひな型として機能するマスターフォームをサポート。もちろん奇数ページ、偶数ページで独立した設定が可能です。

●ノンブル、日付、時刻の印刷
マスターフォーム中にノンブル(ページ番号)、日付、時刻を様々な形式でスタンプすることができます。

●袋とじ印刷のサポート
もちろん、編集時もWYSIWYGで編集可能です。

●CD-ROM辞書検索機能を付加
SX広辞苑付属のLightWing.X(EPWING(V1) CD-ROM簡易検索用シャーペン外部コマンド)をワープロパックにも添付。機能限定版CD-ROMドライバも付属します。

■動作環境
・SX-WINDOW Ver3.1以上
・空きメモリ300KB程度

■付録
・シャーペン外部コマンド開発キット(ライブラリおよびリファレンス)
・IFM ver 4.1
・CD-ROM Driver Ver.2.1.(機能限定版)
・EPWING(V1)CD-ROM簡易検索用シャーペン外部コマンド LightWing.X

Ver.1.0ユーザーの皆様へ大切なお知らせ

Ver.1.0ユーザーの皆様を対象に、バージョンアップサービスを行います。

バージョンアップをご希望の方は、お名前、ご送付先を記入した紙を同封の上、現金封筒にて手数料¥5,000(送料・消費税込み)を下記までご送付ください。

〒321
栃木県宇都宮市竹林町503番地1
(株)計測技研　「ワープロパックバージョンアップ係」

なお、本サービスでお送りするディスクはバージョンアップ専用ディスクです。ワープロパック本体をお持ちでない場合はご使用できませんのでご注意ください。

SX-WINDOW用CD-ROM辞書検索ソフト

SX広辞苑《EPWING対応版》

標準価格 ¥19,800　岩波書店「広辞苑第4版」CD-ROM版バンドルセット¥43,800

・豊富でパワフルな検索方法により、必要な情報をすばやくピックアップ。
・広辞苑の最新版である第4版をもとにしたCD-ROMを使用するので、よりコンテンポラリーなキーワードにアクセス可能です。
・シャーペンと融合して語句の検索を行なうシャーペン用外部コマンド"LightWing.X"を同梱。複雑な検索を行なう場合はSX広辞苑.Xを、普段よく使う単純な検索にはLightWing.Xを、という使い分けも可能です。
・広辞苑第4版CD-ROM版と同様に、EPWING(V1)規約にもとづいたCD-ROMタイトルなら、ほとんどのCD-ROMの内容を検索できます。

●動作環境
・SX-WINDOW 3.0以上
・SX-WINDOW動作中の空きメモリとして1MB以上を推奨
・CD-ROMドライブ(CD-ROM Driver Ver2.0が付属するので、CD-ROM Driverを別途お買い上げいただく必要はありません。CD-ROM Driverのマニュアルや添付ソフト等は付属しません)

68040搭載アクセラレータ

X68040turbo

標準価格¥98,000　ヒートシンク別売¥1,000

040turboは、68040を搭載したX68030(5インチタイプ)専用のアクセラレータです。040turboを装着することで得られるパフォーマンスは、従来の2〜3倍! 計算、特に浮動小数点演算中心のソフトならば、さらにそれ以上の高速化も望めます。
詳しくはソフトバンク刊「X68040turbo〜A Story of Makeing "After X68030"〜」(BEEPs著)をご覧ください。

X680x0用Ether net接続パック

EthernetStarterPack/X680x0

標準価格 ¥88,000

ESP/Xは、Ether netアダプタ「Ether+」と、TCP/IPドライバ、そして基本的なアプリケーションからなるパッケージです。
ftp、telnet(いずれもクライアント)等、基本的なアプリケーションを標準添付。ドライバを活用するためのライブラリも付属します。
※10BASE-2対応モデル・10BASE-T対応モデルの2種類があります。

●動作環境
・Human68k ver3.0以上
・メモリ常駐量500KB前後
・SCSIインターフェース内蔵機種以外はSCSIボードが必要

SCSI-2対応CD-ROMドライブ専用ドライバ

CD-ROM Driver Ver2.10

標準価格 ¥4,800

X680x0用フ…CD-ROM

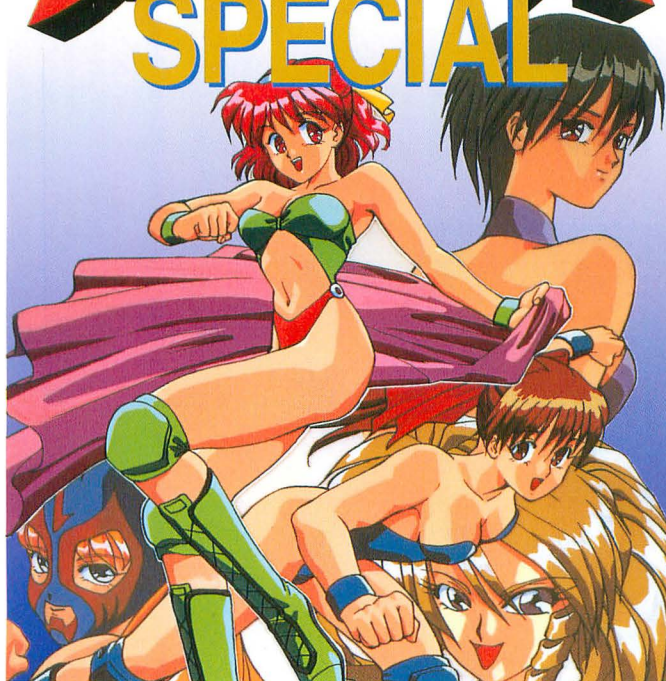
FreeSoftwareSele… 2

完売御礼

お求めはお近くのパソコンショップ、または当社通販部(TEL:0286-22-9811)へお申し込みください。
通販ご希望の方は、ソフト代金+送料¥1,000に消費税を加え、ご住所・お名前・電話番号・商品名を明記した紙を同封の上、現金封筒でお申し込みください。

低金利クレジット　通信販売送料　全国一律¥1,000　長期クレジット可能　※表示価格に消費税は含まれておりません。

株式会社 計測技研　マイコンショップ BASIC HOUSE 本社／ショールーム／通販部
〒321 栃木県宇都宮市竹林町503-1
TEL 0286-22-9811　FAX 0286-25-3970

サポートネット TECOSYS-3 24時間稼動中! (0286)51-1430 (9600bps MNP5)

セクシーでパワフルな女子プロを制覇しろ！

18禁版

カードバトルにプロレスを融合させた、「レッスルエンジェルス」シリーズ。いよいよ最大のヒット作「レッスルエンジェルススペシャル」が登場です。さまざまなイベントの選択によって運命が変わる、マルチシナリオ・マルチエンディング。プロレス技数、カテゴリーが増加して、レスラーの個性もパワーアップ。そして、「恐怖の水着はぎデスマッチ」もパワーアップして復活！18禁だから、そのセクシー度はもうケタ違い！待望のX68000移植完成！明日のトップイベンターを目指すのだ！

機能アップ！

- ●オリジナルオープニングを収録
- ●画面のレイアウトを変更
- ●エキジビションモードグラフィック描き直し
- ●256色モードと16色モードを搭載
- ●サウンドも明るめに変更
- ●AD-PCMによる効果音
- ●ディスクアクセスを最少に抑える設計

このソフトは、全国のパソコンショップで、パッケージ版で販売いたします。TAKERUでは販売致しません。TAKERU事務局では通信販売はいたしませんので、悪しからずご了承下さい。

対応機種：X68000/X68030
要メモリ2Mバイト
（ハードディスク対応）

制作：グレイト

¥8,800（税別）

三國志

知力の極限に挑む、君主、武将、軍師の膨大なデータ。小説よりリアルと、名作の誉れ高い中国統一ゲーム。この歴史的な傑作シリーズはどのようにして始まったのか?SLGファンなら絶対に見逃せない!!

制作/光栄
対応機種/X68000（30不可）　**¥5,200**

三國志 II

登場人物350余名、最大11人まで同時プレイ可能、6編のマルチシナリオ方式、埋状の毒・駈虎呑狼等のユニークな計略要素導入、さらに深みを増した外交・HEX戦など、まさに名作!カシオペアの向谷　実のBGMも話題に。

制作/光栄
対応機種/X68000（30不可）　**¥4,900**

大航海時代

リコエイションゲームシリーズの傑作。毎回違った展開が楽しめるイベントジェネレーティングシステム。帆船の特徴が活かされたHEX戦。失われたロマンを求めて、冒険者たちの航海の旅が始まる。

制作/光栄
対応機種/X68000（30可）　**¥3,400**

維新の嵐

坂本龍馬が、西郷隆盛が、吉田松蔭が日本を憂い、改革を目指して奮い立つ!幕末の志士の個性を際だたせる緻密なパラメータ。出会いの楽しさ、駆け引きを楽しむ新システム。強力な機能で、維新を操れ!

制作/光栄
対応機種/X68000（30不可）　**¥3,400**

信長の野望　戦国群雄伝

400余名の群雄が割拠する下剋上の乱世。配下の羽柴秀吉、柴田勝家を個性豊かな武将たちを、思いのままに操って、戦雲たなびく戦場へ、天下分け目の決戦に臨む！光栄の代表作「信長の野望」シリーズの傑作！

制作/光栄
対応機種/X68000（30可）　**¥3,400**

伊忍道　打倒信長

1つのゲームでSLG とRPG、2つのジャンルが楽しめるリコエイションゲームの第3弾。特にRPGの要素が濃い、異色傑作だ!意志を持ったキャラクターが目的に向かって行動を展開。敵を倒して腕を上げ、技を磨いて信長を倒せ!

制作/光栄
対応機種/X68000（30不可）　**¥3,400**

太閤立志伝

裸一貫の足軽頭から身を興し、関白にまで登り詰めた男・木下藤吉郎(豊臣秀吉)。草履を温めたエピソード・奇跡の墨俣一夜城など、数々の逸話を持つ男の一生を再現する、リコエイションゲームの傑作です。

制作/光栄
対応機種/X68000（30不可）　**¥3,400**

蒼き狼と白き牝鹿　元朝秘史

光栄歴史三部作の一角を成す、草原の英雄チンギス・ハーン。稀代のスケールと空前絶後の迫力で、一代帝国を築き上げた男の豪快な一生を見事に再現!熱いシミュレーションの傑作です。

制作/光栄
対応機種/X68000（30不可）　**¥3,400**

ロイヤルブラッド

新シリーズ「イマジネイションゲーム」のデビュー作。イシュメリアという架空の島国を舞台にした、幻想世界のシミュレーションゲームだ。あなたは独立貴族のひとりとなり、領主達が持っている6つの宝石を集め、イシュメリアの新王となれ!

制作/光栄
対応機種/X68000（30可）　**¥2,700**

ヨーロッパ戦線

戦乱のヨーロッパ。砂塵の彼方から迫り来る黒い車体は、敵か味方か?次々に飛び込んでくる情報、時事刻々と変わる戦局。多彩な兵器やユニット、人間的要素を重視した各種パラメータ。WWIIシリーズ第2弾。勝利の旗を手に入れろ!

制作/光栄
対応機種/X68000（30可）　**¥4,500**

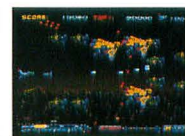
大戦略 III '90

90年代にふさわしくパワーアップされた「大戦略」シリーズ。戦略思考ルーチン、ゲームスピード、コマンド体系、リアルタイムオペレーションなど大幅革新された作品です。

制作／システムソフト
対応機種／ X68000　**¥2,500**

ジェノサイド 2

あのズームのゲームがついに名作文庫に登場！特大キャラとハデハデな演出で、68ユーザーのどぎもを抜いた名作アクションゲームだ。MIDIにも対応しているぞ。

制作/ズーム
対応機種/X68000（30不可）　**¥2,500**

ファランクス

デカキャラ・派手め演出の横スクロールオアワーシューティング。拡大・回転・縮小・多関節・半透明・ラスタースクロール・MIDIと、各種要素がいっぱい詰まってます。

制作／ズーム
対応機種／X68000（30不可）　**¥2,500**

A列車で行こう II

かの「A列車」シリーズの第２弾。パズル的要素がアツクなる!鉄道会社社長の立場で、線路の敷設・撤去を行い、ワールドワイドにマップを発展させていこう。

制作／アートディンク
対応機種／X68000（30不可）　**¥3,800**

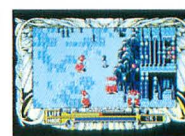
A III（A列車で行こう3）

さらにワイドに、さらに完成度の増した、世界レベルヒットの第３弾。世にA.III ブームを巻き起こしたことで、記憶に新しい超有名作、ついに文庫に登場！

制作／アートディンク
対応機種／X68000（30可）　**¥3,800**

栄冠は君に

高校野球シミュレーションシリーズの、記念すべき第１作。全国制覇を達成するには、3990校の頂点に立たなければならない。感動の優勝セレモニーを、果たして見ることが出来るか！？

制作／アートディンク
対応機種／X68000　**¥3,800**

ルーンワース「黒衣の貴公子」

ハイドライドシリーズに続く、新ARPGシリーズ第1弾。綿密に構築された世界「ルーンワース」を舞台に、極めて自由度の高いゲームシステムの中で、興奮の冒険が始まります。

制作／T&Eソフト
対応機種／X68000　**¥ 700**

イース III（ワンダラーズフロムイース）

よりアクション性を増した、これまた、大人気を博したアクション・ロールプレイング。アドルの最後の冒険物語でした。攻撃方法もいっそう多彩になって、時間を感じさせない逸品です。

制作／日本ファルコム
対応機種／X68000（30不可）　**¥2,000**

パソコンソフト自動販売機
TAKERU

TAKERU事務局
〒467　名古屋市瑞穂区苗代町2番1号
ブラザー技術開発センタービル2F
TEL(052)824-2493（受付時間：月〜金　13:00〜18:00）

営業所
東京営業所
(03) 5443-4967
大阪営業所
(06) 258-3024

通信販売 1994年4月1日より、送料／手数料が有料になりました。

ソフト名、機種名、メディアのサイズ、住所、氏名、電話番号を明記の上TAKERU事務局まで現金書留でお申し込みください。送料／手数料は、1回のお申し込み総金額が5,000円以上の方は無料。4,900円までの方は500円をいただきます。4,900円までの方は現金500円をプラスしてお申し込みください。誠に勝手ながら、皆様のご理解とご協力の程、お願い申し上げます。

T1002179090768 雑誌 02179-9